MODELING OF TRANSISTOR SELF-HEATING FOR CIRCUIT SIMULATION

By

DAVID T. ZWEIDINGER

A DISSERTATION PRESENTED TO THE GRADUATE SCHOOL
OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

UNIVERSITY OF FLORIDA

1997

ACKNOWLEDGMENTS

I would like to thank Dr. Robert Fox for his patience, the Semiconductor Research Corporation for the financial support, the many people of Harris Semiconductor for their assistance and encouragement, Jon Brodsky for his long hours simulating heat-flow, and finally my parents for doing it all before me and reminding me why it was worth the work.

TABLE OF CONTENTS

page

ACKNOWLEDGMENTS ii

ABSTRACT v

CHAPTERS

1 INTRODUCTION 1

2 BACKGROUND 4

The SPICE Gummel-Poon Model 4
Thermal Impedance Modeling 6
Self-Heating Models 10

3 MODELING FOR CIRCUIT SIMULATION 15

An Overview of SPICE 15
Formulation of the Self-Heating Model 17
Implicit Temperature Approach 18
Thermal Impedance Modeling 19
The DC Solution 20
AC Modifications 27
Advantages and Limitations 28
Explicit Temperature Node Approach 29
The DC Solution 29
Implementation Options 32
SPICE Code Modification 34
Advantages and Limitations 39
Convergence 40

4 SELF-HEATING IN CIRCUITS 44

Self-Heating Effects 44
Simulation Examples 46
Convergence Comparison 57

5 MEASUREMENT OF MODEL PARAMETERS 60

Measurement of the Thermal Impedance 60
Temperature Coefficient Measurement 62
Thermal Impedance Measurement 67
Thermal Impedance Model and SPICE 72
Self-heating and Model Parameter Extraction 75

6 SELF-HEATING FOR SOISPICE 83

Self-Heating Implementation 85
The SOI MOSFET Model 85
Specifying Self-Heating 86
Computing the Derivatives 87
Computing the Power Dissipation 88
Local Temperature Update of Parameters 88
Loading the Jacobian 90
Convergence Issues 92
Initial Simulation Results 93
Conclusion 97

7 CONCLUSION 98

APPENDIX A 101

APPENDIX B 115

APPENDIX C 127

APPENDIX D 140

BIBLIOGRAPHY 148

BIOGRAPHICAL SKETCH 153

Abstract of Dissertation Presented to the Graduate School
of the University of Florida in Partial Fulfillment of the
Requirements for the Degree of Doctor of Philosophy

MODELING OF TRANSISTOR SELF-HEATING
FOR CIRCUIT SIMULATION

By

David T. Zweidinger

August 1997

Chairman: Robert M. Fox
Major Department: Electrical and Computer Engineering

Semiconductor devices consume power during operation. This power is dissipated mostly in the form of heat that increases the temperature of the device. This effect is known as self-heating. For bipolar transistors, BJTs, and silicon-on-insulator metal-oxide-semiconductor field-effect transistors, SOI MOSFETs, self-heating can significantly change device performance. Circuit designers need an easy way to quantify the performance changes from self-heating, within the compact models used in today's circuit simulators.

A historical review of other attempts at modeling self-heating in circuit simulators is shown. Two approaches to including self-heating effects for bipolar transistors in the popular circuit simulator SPICE are presented. One approach uses an implicit temperature and transcendental equations to correct for self-heating in the device model. The other

approach uses an explicit temperature and an electrical subcircuit analogy for power generation and thermal dissipation to model self-heating. Techniques for assessing self-heating effects in hand calculations are shown, and example simulations with self-heating in common dc, ac, and transient circuits are shown. Reduced amplifier gain and thermal mismatching are common self-heating effects.

A new method for extracting the thermal impedance of bipolar transistors is derived, and results are shown for transistors of several different geometries. The effects of self-heating on the extraction of the Gummel-Poon model parameters are explored using long-time measurements and the modified SPICE in the optimization process.

The implementation of self-heating in the physics-based SOISPICE model for SOI MOSFETs is shown. Common circuit examples are simulated showing the effects of self-heating. For analog circuits, the negative temperature coefficient of drain current can cause problems for biasing and amplifier gains. For digital circuits, self-heating is a function of frequency and is not significant.

CHAPTER 1
INTRODUCTION

In present circuit simulators, temperature is primarily modeled globally. Only a single temperature is used for all of the circuit elements, and the circuit is simulated isothermally at that temperature. This approach ignores self-heating, the local increase in temperature from the power dissipation of a device. The operating point of a device defines its power generation, and the geometries and materials affect how the power is dissipated as heat. The isothermal simplification is inaccurate considering that most circuits have many elements of different geometries, operating under different bias, and dissipating different amounts of power. The result is a performance change of the circuit because of all of elements operating at different temperatures. These performance changes from self-heating are significant for many circuits.

Semiconductor devices are especially sensitive to their operating temperature. One device that is particularly sensitive is the bipolar junction transistor (BJT). In BJTs the dominant sensitivity arises from the exponential dependence of the carrier concentration on temperature. Only a few degrees Kelvin of temperature change will cause significant changes in the carrier concentration. The BJT terminal currents are a function of the carrier concentration, and hence are greatly affected by

small temperature changes. Some temperature dependent mechanisms that are weaker than the carrier concentration are the carrier mobility and recombination lifetimes which become significant when large temperature changes occur on the order of tens of degrees Kelvin. These two mechanisms are important to silicon-on-insulator metal-oxide-silicon field-effect transistors, SOI MOSFETs. Self-heating can cause failure of heterojunction bipolar transistors (HBT) where the large negative temperature coefficients cause currents to drop to zero in relatively low-power operation [Gao89]. Including these effects in the compact BJT and SOI MOSFET models used in circuit simulation is important for improving accuracy in some circuits.

The importance of self-heating has become more evident with changes in technology. The continued shrinking of geometries, and the resulting increase in current densities and restriction of heat flow, tend to increase the thermal impedance and raise device operating temperatures. The thermal impedance is the temperature increase for a given power dissipation. The trend toward the use of silicon dioxide, SiO_2, for isolation makes self-heating more important, because the decreased thermal conductivity of SiO_2 compared to silicon creates a barrier to heat flow and greatly increases the thermal impedance of the device.

Finite element-based simulators are available that can simulate a BJT under dc and ac conditions with self-heating [Lia94], but these are extremely slow for analysis and design of circuits with multiple devices.

What circuit designers need is a compact modeling simulator that includes self-heating. SPICE, with its Gummel-Poon model, is a popular circuit simulator. The temperature dependencies of all the parameters are implemented in the code for simulation of circuits at different global temperatures. The goal is to implement self-heating in the model by incorporating the temperature dependencies and the device thermal impedance and power dissipation to allow each device to be simulated at its own local temperature.

The following chapters are organized to give a understanding of the motivations for this work and the solutions reached for incorporating self-heating into circuit simulation. Chapter Two a historical view of thermal impedance modeling and self-heating simulation. Chapter Three describes two ways to incorporate self-heating into SPICE. Chapter Four focuses on circuit analysis in the presence of self-heating and shows some example simulations. Chapter Five shows a new extraction method for measuring thermal parameters of BJTs, in particular the thermal impedance. The effects of self-heating on model parameter extraction are also examined. Chapter Six shows the implementation of a self-heating model for SOI MOSFETs, with some simulation examples included. Finally, Chapter Seven summarizes the information and insights presented.

CHAPTER 2
BACKGROUND

This chapter presents an overview of previous work in modeling self-heating. First, the BJT device model equations used as the framework for self-heating simulation are reviewed. Following this is an investigation of thermal impedance modeling. Finally, an examination of previous attempts to model self-heating in BJTs will show the need for improvement.

The SPICE Gummel-Poon Model

The dominant model for use in simulating bipolar transistors in circuits was introduced in 1970 [Gum70]. This integral charge-control model was incorporated into the circuit simulator SPICE, developed at the University of California at Berkeley. Most circuit simulators in use today have a core based on SPICE and its Gummel-Poon BJT model. Therefore, SPICE was the obvious platform for the study of self-heating in BJTs and circuits. The communication of self-heating modeling is helped by the common use of SPICE based simulators in industry. What follows is a short overview of the SPICE Gummel-Poon model that will form the basis for most of the discussion in the chapters to come.

The following is the Gummel-Poon model for an npn BJT. The base current (I_B) is composed of four current elements,

$$I_B = I_{BF} + I_{BE} + I_{BR} + I_{BC}, \tag{2.1}$$

where I_{BF} is the forward diffusion current that results from back-injected holes across the base-emitter junction, given by

$$I_{BF} = \frac{I_S}{\beta_F}\left(e^{\frac{V_{BE}}{N_F V_T}} - 1\right). \tag{2.2}$$

where I_S is the saturation current, β_F is the forward dc current gain, N_F is the forward current emission coefficient, and V_T is the thermal voltage. The non-ideal base-emitter current results from recombination in the quasi-neutral base, and is given by:

$$I_{BE} = I_{SE}\left(e^{\frac{V_{BE}}{N_E V_T}} - 1\right) \tag{2.3}$$

where I_{SE} is the base-emitter leakage saturation current and N_E is the leakage emission coefficient. Similarly, the dual for the reverse-bias case exists. I_{BR} is the reverse diffusion current given by:

$$I_{BR} = \frac{I_S}{\beta_R}\left(e^{\frac{V_{BC}}{N_R V_T}} - 1\right) \tag{2.4}$$

where β_R is the reverse dc current gain and N_R is the reverse current emission coefficient. Finally the non-ideal base-collector current is

$$I_{BC} = I_{SC}\left(e^{\frac{V_{BC}}{N_C V_T}} - 1\right) \tag{2.5}$$

where I_{SC} is the base-collector leakage saturation current and N_C is the leakage emission coefficient.

The collector current is given by

$$I_C = \frac{I_S}{Q_B}\left(e^{\frac{V_{BE}}{N_F V_T}} - e^{\frac{V_{BC}}{N_R V_T}}\right) - I_{BR} - I_{BC} \tag{2.6}$$

where Q_B is the base charge factor that accounts for base-width modulation and high-injection conditions. The base charge factor is

$$Q_B = \frac{Q_1}{2}(1 + \sqrt{1 + 4Q_2}) \tag{2.7}$$

where

$$Q_1 = \left(1 - \frac{V_{BC}}{V_{AF}} - \frac{V_{BE}}{V_{AR}}\right)^{-1} \tag{2.8}$$

and

$$Q_2 = \frac{I_{SE}}{I_{KF}}\left(e^{\frac{V_{BE}}{N_F V_T}} - 1\right) + \frac{I_{SC}}{I_{KR}}\left(e^{\frac{V_{BC}}{N_R V_T}} - 1\right). \tag{2.9}$$

The parameters V_{AF} and V_{AR} are the forward and reverse early voltages, and I_{KF} and I_{KR} are the forward and reverse knee currents or corners for beta high current roll-off.

Thermal Impedance Modeling

As a device heats from its power dissipation, its operating point shifts causing the power dissipation to change. This circular mechanism of heating is called thermal feedback. There are several thermal feedback

mechanisms in integrated circuits, which can usually be considered separately [Mei77, Fox91c]. One mechanism is the rise in the overall chip temperature due to the total power dissipated on the chip. This temperature rise is controlled by the chip-to-package and package-to-ambient thermal impedances. This global heating operates over a long time scale (milliseconds to minutes), and couples all of the devices on a chip. Global heating can generally be reduced by careful packaging and heat-sinking, and is accounted for in circuit simulation by simulating at multiple temperatures. Global heating is important to recognize in parameter extraction, where long thermal settling times cause the isothermal assumption to become inaccurate.

For large-area devices, or those dissipating large power, there can be direct coupling between the heat dissipated in one device and the temperature of other devices. This mechanism is strongly affected by the circuit layout. For example, thermal coupling in an operational amplifier from the output stage back to the input can profoundly affect the circuit's gain characteristics, although careful layout exploiting symmetry can mitigate these effects [Fuk76].

Direct heating of a transistor by its own power dissipation cannot be eliminated through packaging, scaling, or layout. This mechanism can cause substantial errors in modeling even without high power dissipation. As shown by Joy and Schlig, for modest power dissipation, the temperature rise is mostly confined within the transistor itself [Joy70].

Thus this effect can be simulated only by allowing each transistor to be modeled at its own temperature, controlled by its thermal spreading impedance and power dissipation. The thermal spreading impedance, which controls the temperature rise, can be predicted from the transistor's geometry [Lee92, Bro93, and Zwe95b].

The thermal impedance model derived by Fox and Lee [Fox91b] is the frequency-domain equivalent of the time-domain derivation shown by Joy and Schlig [Joy70], and is based on the same assumptions. All of the heat is assumed to be generated uniformly in the rectangular volume of the collector space-charge region (SCR). The analysis assumes an adiabatic plane at the surface of the transistor created by an image of the SCR heat-source placed above the surface. This approach assumes there is little heat lost through the top of the BJT and the heat is removed predominately by the conduction of the body of the semiconductor. For isolation technologies where SiO_2 is used between devices, this may not be a good assumption [Dav92, Bro97a]. In the time domain, the response to a point impulse of heat, measured a distance r away in a uniform medium, is

$$T(r,t) = \frac{\kappa}{8K(\pi\kappa t)^{3/2}}\exp\left(-\frac{r^2}{4\kappa t}\right) \tag{2.10}$$

where K is the thermal conductivity and κ is the thermal diffusivity. The Laplace transform of (2.10) gives the thermal impedance

$$Z_{TH}(r,s) = \frac{1}{4\pi Kr}\exp\left(-r\sqrt{\frac{s}{\kappa}}\right) \tag{2.11}$$

For $s = 0$, this equation gives the thermal resistance R_{TH} for the point source.

An expression for R_{TH} at any point r' = (x', y', z') in the semiconductor due to the heat generated in the collector SCR can be derived by integrating R_{TH} over the collector SCR and its image source. For a rectangular vertical BJT the integral is

$$R_{TH}(W, L, D, H, r') = \frac{1}{4\pi K}\int_0^W \int_0^L \left[\int_D^{(D+H)} \frac{dxdydz}{r} + \int_{-D}^{-(D+H)} \frac{dxdydz}{r}\right] \qquad (2.12)$$

where $r = \sqrt{(x-x')^2 + (y-y')^2 + (z-z')^2}$, W and L are the width and length of the emitter, D is the collector-base junction depth, and H is the SCR width. The result in (2.12) can be numerically approximated by [Fox91a, Fox91b]:

$$R_{TH} = \frac{1}{4\pi K\sqrt{WL}\cdot f_1 \cdot f_2} \qquad (2.13)$$

where

$$f_1(d, h) = (0.058d + 0.14)h + 0.34d + 0.28 \ , \qquad (2.14)$$

$$f_2(a) = 0.98 + 0.043a - 6.9\cdot 10^{-4}a^2 + 3.9\cdot 10^{-6}a^3 \ , \qquad (2.15)$$

$$d = \frac{D}{\sqrt{WL}}, \qquad (2.16)$$

$$a = \frac{W}{L}, \ a > 1 \ , \qquad (2.17)$$

$$h = \frac{\sqrt{2\varepsilon(V_{CB} + \phi_{bi})/(qN_{EPI})}}{\sqrt{WL}} \ , \qquad (2.18)$$

N_{EPI} is the epitaxial doping density in atoms/cm^3, V_{CB} is the collector-base junction voltage and ϕ_{bi} is the collector-junction built-in potential. This expression was found experimentally to approximate R_{TH} well for a wide

variety of rectangular emitter geometries [Fox91a].

Based on a comparison of (2.11) and (2.13), the same R_{TH} could be computed by replacing the collector SCR with a point heat source placed a distance r_{eff} below the emitter, where r_{eff} can be defined as

$$r_{eff} = \frac{1}{2\pi \cdot K \cdot R_{TH}} = 2\sqrt{W \cdot L} \cdot f_1 \cdot f_2. \tag{2.19}$$

This r_{eff} can be used in (2.11) to simply estimate for Z_{TH} for all frequencies as

$$Z_{TH} = R_{TH} \cdot \exp\left[-r_{eff}(1+j)\sqrt{\frac{\omega}{2\kappa}}\right] \tag{2.20}$$

Equations (2.19) and (2.20) can also be used to model Z_{TH} based on a measured value of R_{TH}.

Figure 2-1 compares a normalized temperature response to the model given by Joy and Schlig [Joy70] and the Fourier transform of (2.20). Note that the point-source representation overestimates the phase of Z_{TH}. However, the magnitude of Z_{TH} is predicted well, and the errors in phase coincide with small $|Z_{TH}|$, so that the overall error is small. Note that $|Z_{TH}|$ varies over a much wider frequency range than would be predicted by a single-pole thermal equivalent circuit, in which $|Z_{TH}|$ would decay primarily over one decade of frequency.

Self-Heating Models

In 1964 Müller proposed modeling self-heating using an electrical subcircuit analog to the power dissipation through the thermal spreading impedance [Mül64]. As shown in Figure 2-2, the power generated in the

device flows as a current through an impedance, establishing a potential equivalent to the change in temperature. Only a single RC pole is shown,

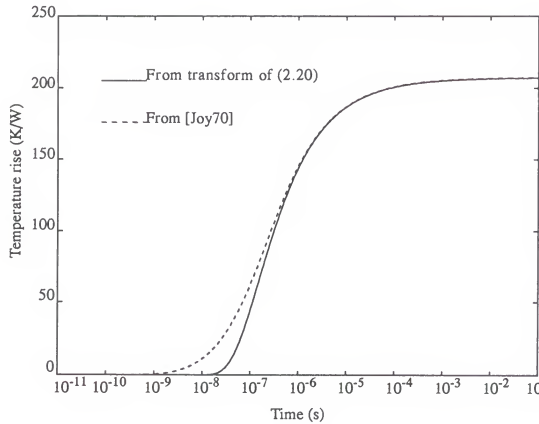

Figure 2-1 Normalized temperature response to a 1 W power step for a 7x10μm npn BJT, using the model in [Joy70] and the inverse Fourier transform of the frequency response of (2.20) [Fox93b].

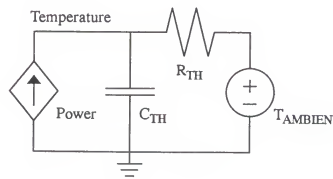

Figure 2-2 Example of thermal subcircuit used to model self-heating.

but electrical networks representing multiple poles could be constructed to better model the distributed nature of the thermal spreading impedance [Mül64].

An early attempt at computer simulation of self-heating effects was done by Latif and Bryant [Lat81]. In this work a simple, single time-constant, thermal subcircuit was implemented into the Ebers-Moll BJT model [Ebe54]. They used a linear network solver (WATAND) to simulate dc and transient effects of second breakdown in power BJTs. Later, this work was extended to include the more accurate Gummel-Poon model in the same solver for a study of self-heating effects on current mirrors [Mun91]. These two works failed to include the effects of self-heating in the ac analysis.

A single-pole thermal subcircuit was used to model temperature in SPICE by Vogelsong and Brzezinski [Vog89]. Results from a few circuit simulations were presented. Again, only dc and transient analysis were implemented, ignoring the important ac analysis. Their approach to computing the power dissipation included heat sources other than the base-collector junction which all have different thermal impedances that were not modeled. The algorithms they used to implement self-heating are vague.

In all of the above implementations, values used for the thermal impedance are empirical. The impedance value used is the one that makes the simulation fit the data best. This approach can neglect the coupling of

self-heating effects to other effects in the device behavior. Most of the implementations used only a single-pole network which poorly models the distributed nature of the thermal impedance. Finally, all of the simulators above require a voltage source in the thermal network to represent the ambient temperature. Repeating a simulation at a different ambient temperature would require modification of all of the thermal networks in the circuit.

The underlying physics of the heat flow either within the transistor or between transistors is often neglected in the above works. Heat-flow analysis shows that except for rather high-power transistors, temperature gradients are mostly confined within the transistor itself [Joy70]. Usually adjacent devices are thermally coupled through the device-to-chip-to-package spreading impedances, that have very long time constants, rather than device-to-device impedances [Lee92]. The heat coupling effects can usually be ignored. Where the heating of adjacent devices cannot be ignored is in multi-fingered emitter BJTs and HBTs. The close proximity of each finger causes a strong thermal coupling. For example, for a three finger BJT driven with a constant current, the middle finger is heated by the two outside fingers causing its current to increase and the others to decrease. This effect can degrade the performance of multi-fingered emitter devices.

The improved implementation shown in the next chapter includes dc, transient, and ac analyses. It allows flexibility in formation of the

thermal impedance network, so that the self impedance and impedance to adjacent devices or emitter fingers can be modeled accurately.

CHAPTER 3
MODELING FOR CIRCUIT SIMULATION

Because of the deficiencies of previous attempts to include self-heating in circuit simulation, a more complete self-heating implementation was needed. The first priority was to create a simulator that could model the ac effects of self-heating. The frequency domain had been mostly ignored for analog circuits. The first implementation filled this void using an implicit local temperature approach. In the implicit temperature approach the BJT model is reduced to a two-port y-parameter equivalent model. The y-parameters are then modified by self-heating, and the modified y-parameters are converted back to the original model form. Unfortunately, this implementation made transient analysis very difficult, so another implementation using an explicit temperature and thermal subcircuit was done. A thermal subcircuit represents the power dissipation, heat flow, and temperature rise. This requires a thermal impedance network with some inherent limitations, but does allow transient analysis, unlike the implicit node approach. Both of these approaches are described in this chapter.

An Overview of SPICE

To understand the implementations, it is important to understand first the organization and function of SPICE. The circuit simulation

program can be broken down into four pieces: pre-processing, dc and transient solution, ac analysis, and post-processing.

The pre-processing section is responsible for reading the circuit description file and loading the information into the data structure. This processing includes building a instance matrix of all the circuit elements to preserve the connectivity information, and associating any parameter values and models that go with each instance. The data structure for the Jacobian matrix, voltage and current vectors, and state variables is then built. Any processing of model parameters is done, and the global temperature dependences are updated for the selected temperature of the simulation.

The dc and transient section of SPICE loads the Jacobian matrix with the circuit conductances and branch currents, and solves the resulting linear equations for the node voltages. For nonlinear elements an iterative Newton-Raphson method is employed. The device models are used in routines to compute the next guess for the conductances, currents, and voltages associated with each device. During transient analysis the energy of storage elements is integrated for each time step, and reduced to an equivalent conductance and current for the matrix load.

The ac analysis section uses the real part of the Jacobian from the solution in the dc analysis, and computes the complex part of the matrix for reactive elements at a given frequency. Perturbation of a node voltage

or branch current gives the small-signal response at the linearized operating point.

Finally, the post-processing routines gather the simulation results and prints or plots the data. The results can include device operating point information, sweep data from ac, dc, and transient analyses, and simulation statistics.

Formulation of the Self-Heating Model

The derivation of the both modeling approaches shown here starts the same way, with the expression for the small-signal collector current. For the current into the collector including self-heating

$$i_c = \left.\frac{\partial I_C}{\partial V_{BE}}\right|_{T.fixed} \cdot v_{be} + \left.\frac{\partial I_C}{\partial V_{CB}}\right|_{T.fixed} \cdot v_{cb} + \frac{\partial I_C}{\partial P} \cdot p \qquad (3.1)$$

where the power is

$$p = I_C v_{ce} + i_c V_{CE} + I_B v_{be} + i_b V_{BE}. \qquad (3.2)$$

Expanding the power term in (3.1) and reducing the partial derivatives to conductances results in

$$i_c = g_{oE} v_{ce} + g_{mE} v_{be} + g_{\mu E} v_{cb} + \frac{\partial I_C}{\partial T} \cdot \frac{\partial T}{\partial P} p \qquad (3.3)$$

where the E subscript denotes electrical-only. Substitution of the thermal resistance gives

$$i_c = g_{oE} v_{ce} + g_{mE} v_{be} + g_{\mu E} v_{cb} + \frac{\partial I_C}{\partial T} \cdot R_{TH} p. \qquad (3.4)$$

The last partial is substituted using the definition of the fractional temperature coefficient given by

$$D_m = (1/I_m)\frac{dI_m}{dT} \tag{3.5}$$

where m is c or b for the collector or base current. The resulting substitution gives

$$i_c = g_{oE}v_{ce} + g_{mE}v_{be} + g_{\mu E}v_{cb} + I_C D_C R_{TH} p. \tag{3.6}$$

Equation (3.6) marks the point of divergence of the two implementation approaches. Solving for i_c gives a ratio with temperature represented implicitly as done by Müller [Mül64]. This is the basis of the implicit temperature approach [Zwe92]. If the last term in (3.6) is represented using temperature for the thermal resistance and power product, a new matrix node is needed. This is the basis for the explicit temperature approach.

Implicit Temperature Approach

This is the simplest modification to make to the SPICE source code. The only changes needed are the addition of the new model parameters and modification of the device model routine. There are no new model matrix elements or state variables that would require changes to the device matrix. Detailed FORTRAN code modifications to SPICE2G.6 for this approach can be found in Appendix A.

Thermal Impedance Modeling

In this approach the thermal impedance is computed for each iteration of the device solution. New model parameters to describe the geometry of the BJT are added to implement the thermal impedance equations discussed in the last chapter. These parameters are: the length and width of the emitter, the depth of the collector-base metallurgical junction, and the collector doping concentration. They are added to the list of values available on the BJT model line. From these four parameters and the collector-base bias voltage the effective thermal resistance is computed from equations (2.13) through (2.18) for each iteration in the dc solution.

For the ac analysis, the effective heat source radius, r_{eff}, is computed from the thermal resistance computed in the dc solution or given in the model line. Then (2.20) is used to compute the thermal impedance at each frequency. The thermal impedance can also be computed from the thermal resistance and capacitance by addition of the thermal capacitance to the model line. This allows a simple single-pole model to describe the thermal impedance.

A hierarchy was formed to manage these three models. The first option is the lowest level where R_{TH} and C_{TH} (thermal capacitance) must be supplied and are used in the dc and ac analyses. The next option is used when only R_{TH} is supplied. In this case, the given R_{TH} is used in the dc solution, and Z_{TH} is computed from r_{eff} for each frequency in the ac

analysis. The last option uses the geometry information and the predictive model to compute R_{TH} for the dc solution and Z_{TH} for the ac analysis.

The DC Solution

The following is a brief description of the changes made to the routine that computes the dc solution for the BJT.

Including the temperature dependence

A first step to all approaches to modeling self-heating is to copy all temperature dependences to the inside of the Newton-Raphson loop. Normally in SPICE the model parameters are updated for temperature once before each simulation is begun, allowing circuits to be sequentially simulated at different global temperatures. By moving the parameter dependences inside the Newton-Raphson loop, the temperature-dependent parameters for each transistor are updated for its local temperature in each iteration. The dependent-parameter temperature functions are described below.

The thermal voltage is implicitly temperature dependent: $V_T = kT/q$ where k is Boltzman's constant and q is the electron charge. The temperature dependence of the dc current gain parameters stem from the dependence of the junction barrier height on temperature, and are given as

$$\beta_F = \beta_{F0}\left(\frac{T}{T_0}\right)^{XTB} \tag{3.7}$$

and

$$\beta_R = \beta_{R0}\left(\frac{T}{T_0}\right)^{XTB} \tag{3.8}$$

where T_0 is the original model parameter temperature and XTB is the β temperature factor. The following current parameters dependences result mostly from the strong temperature dependence of the carrier concentration:

$$I_S = I_{S0} \cdot e^{factln}, \tag{3.9}$$

$$I_{SE} = \frac{I_{SE0}}{(T/T_0)^{XTB}} \cdot e^{\frac{factln}{N_E}}, \tag{3.10}$$

and

$$I_{SC} = \frac{I_{SC0}}{(T/T_0)^{XTB}} \cdot e^{\frac{factln}{N_C}}. \tag{3.11}$$

The exponential factor in the above equations is

$$factln = \frac{E_G}{V_{T0}} - \frac{E_G}{V_T} + XTI \cdot \ln\left(\frac{T}{T_0}\right) \tag{3.12}$$

where XTI is saturation current temperature factor and

$$E_G = E_{G0} - E_{GAP} \cdot \frac{T^2}{T + T_{GAP}} \tag{3.13}$$

is the semiconductor energy gap. Values for the arguments in (3.13) are shown in Table 3-1 for common substrate materials. The SPICE BJT model assumes the energy gap is constant and uses a single parameter in the model line to assign a value. This has not changed for this implementation. Equation (3.13) is added to the temperature equations for the explicit node approach

Material	E_{G0} (eV)	E_{GAP} (eV/K)	T_{GAP} (K)
Silicon	1.16	7.02×10^{-4}	1108
Germanium	0.67	4.56×10^{-4}	210
Gallium Arsenide	1.52	5.41×10^{-4}	204

Table 3-1 Energy gap temperature modeling parameter values for common device materials.

to fix this deficiency in SPICE. New model parameters to assign values for the coefficients were also added to the model line.

Computing the temperature

The temperature is not represented in the Jacobian in this approach. Instead, it is computed in each iteration from the thermal resistance and the power dissipation, $T = P \cdot R_{TH}$. The power is computed in each iteration of the Newton-Raphson loop from the dissipation in the base-collector and base-emitter space-charge regions, $P = I_C V_C + I_B V_B$. Note this is the real power dissipation, and does not include any reactive power that may be present in a transient simulation. The thermal resistance is either provided by the user or computed from the geometry using equations (2.13) to (2.18).

Modifying the Jacobian

In the dc analyses the hybrid-π parameters are converted into two-port y-parameters using

$$Y_{11E} = g_\pi + g_\mu \tag{3.14}$$

$$Y_{12E} = -g_\mu \tag{3.15}$$

$$Y_{21E} = g_m - g_\mu \tag{3.16}$$

$$Y_{22E} = g_o + g_\mu \tag{3.17}$$

where the hybrid-π parameters are shown in Figure 3-1.

Next the BJT common-emitter y-parameters are corrected to include thermal feedback using [Mül64]

$$Y_{mn} = \frac{Y_{mnE} + D_m R_{TH} I_m I_n}{1 - D_m R_{TH} P} \tag{3.18}$$

where m or n equal 1 for the base or 2 for the collector, Y_{mnE} is the uncorrected electrical y-parameter, and D_m is the fractional temperature coefficient of the base or collector current. The collector current temperature coefficient in the forward-active region of operation is estimated by [Zwe93]

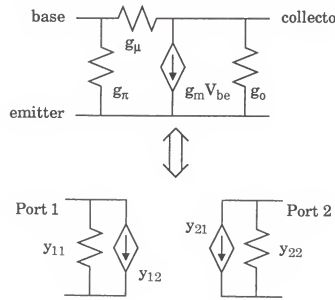

Figure 3-1 Conversion between standard hybrid-π and two-port y-parameter models.

$$D_C = f_1\left(1 - I_C\frac{q_1}{I_{KF}\sqrt{1+4I_{BE}/I_{KF}}}\right) \tag{3.19}$$

where

$$f_1 = \frac{XTI + qE_G/kT - qV_{BE}/kTN_F}{T} \tag{3.20}$$

For the base current temperature coefficient in same region the expression is

$$D_B = \left[I_{BE}\left(\frac{f_2}{N_E} - \frac{V_{BE}q}{kTN_E} - XTB\right) + I_{bf}\left(f_2 - \frac{V_{BE}q}{kTN_F} - XTB\right)\right]/(I_B T) \tag{3.21}$$

where

$$f_2 = XTI + \frac{E_G q}{kT}. \tag{3.22}$$

The thermally corrected y-parameters are then converted back to the hybrid-π form using the following ordered equations:

$$g_\mu = -Y_{12} \tag{3.23}$$

$$g_\pi = Y_{11} - g_\mu \tag{3.24}$$

$$g_m = Y_{21} + g_\mu \tag{3.25}$$

$$g_0 = Y_{22} - g_\mu \tag{3.26}$$

The hybrid-π parameters are then loaded into the admittance matrix. Again, the advantage here is that no new matrix elements are needed in the Jacobian to compute the self-heating effects.

Flow control

The flowchart of the dc solution routine with self-heating is shown in Figure 3-2. First, the voltages from the Jacobian matrix solution are loaded, and the currents are linearly predicted using the conductances, the change in node voltages from the previous iteration, and the previous iteration currents. Using these predicted values of current and the node voltages, the power is computed, followed by the temperature using R_{TH}. (If the predictive R_{TH} model is used then the value for R_{TH} comes from the previous iteration.)

Next, the convergence of the currents, voltages, and temperature is checked. The predicted currents and the node voltages are compared to the values of the previous iteration, and if each comparison is within its specified tolerance, the device has converged. If the convergence check succeeds then the state-variables are saved, the BJT is marked as converged and the model routine is exited. If the convergence criteria are not met then the predicted temperature is used to update the parameter temperature dependences and the currents are computed from the model equations described in Chapter 2.

With the currents computed from the model, the power, R_{TH}, the temperature, and the fractional temperature coefficients are computed. The convergence is checked again for the new current values against the predicted values computed earlier. If the convergence check succeeds then the state variables are saved and the BJT is marked as converged. If it

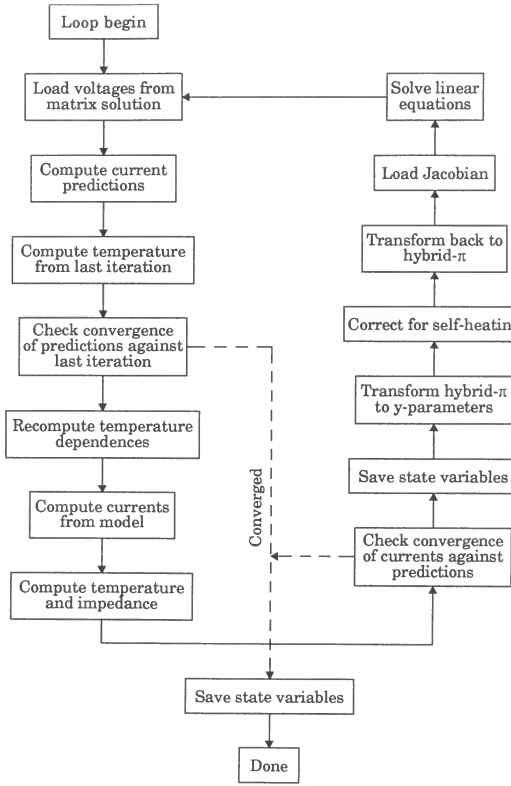

Figure 3-2 Flowchart of the dc solution for the implicit node self-heating approach.

fails the state-variables are saved and the matrix load begins. The BJT conductance values are transformed into y-parameters using (3.14) to (3.17), corrected for self-heating using (3.18), and converted back to hybrid-π form using (3.23) to (3.26). The modified conductances are then loaded into the Jacobian. After all the BJTs in the circuit have been processed and loaded into the Jacobian, the linear equation solver is invoked to generate a new voltage solutions. SPICE continues to iterate until all the nonlinear devices converge, or a maximum iteration count is reached.

AC Modifications

For the ac analysis the small-signal power is computed from (3.2). The hybrid-π model is transformed into complex y-parameters as follows:

$$Y_{11E} = g_\pi + g_\mu + j\omega(C_{xcb} + C_{be} + C_{bc}) \tag{3.27}$$

$$Y_{12E} = -g_\mu + j\omega(C_{xcb} + C_{bc}) \tag{3.28}$$

$$Y_{21E} = g_m - g_\mu + j(xgm - \omega C_{bc}) \tag{3.29}$$

$$Y_{22E} = g_o + g_\mu + j\omega \cdot C_{bc} \tag{3.30}$$

where C_{be} and C_{bc} are the base-emitter and base-collector capacitances respectively, xgm is the excess phase term, C_{xcb} is a base-emitter transcapacitor controlled by the base-collector voltage, and the other terms are standard hybrid-π parameters shown earlier.

The complex form of (3.18) where Z_{TH} is substituted for R_{TH} is used to modify the y-parameters to include self-heating. The fractional

temperature coefficients used are from the dc solution. The thermal impedance is computed from R_{TH} and C_{TH} given by the user or predicted from the R_{TH} computed in the dc solution.

The real parts of the y-parameters are transformed back to the real parts of the hybrid-π model much as in the dc analysis. To transform the imaginary parts, first (3.30) is solved for C_{bc}. Then C_{xcb} and xgm are solved from (3.28) and (3.29). Finally, C_{be} is solved from (3.27). After transforming the modified y-parameters back to the original parameter set, the Jacobian is loaded and the matrix is solved for the current frequency.

Advantages and Limitations

This implementation is easy to use; with only a few new model parameters the user can model self-heating. The changes to the original SPICE source code are minimal as well, making the model easy to support. The bias- and frequency-dependent thermal impedance model makes for smooth, accurate dc and ac simulations.

The self-heating model in this implementation is only valid in the forward-active region of operation. The most significant self-heating occurs in the forward-active region, where the collector-base junction has a large reverse-bias voltage and the high collector current create large power dissipation. The forward-active region is also the most common region for analog applications where self-heating has the greatest effect on

circuit behavior. It is for these reasons that this implementation has focused on this region.

The greatest drawback to this implicit temperature approach is the inability to do transient simulations. SPICE is ill-equipped to solve the problem for this modeling approach, and this is the ultimate reason why the explicit thermal node method was implemented.

Explicit Temperature Node Approach

In the previous approach, the temperature was computed implicitly in each iteration using the voltage and currents of each iterate, and then modifying the existing internal conductances. In the following approach, temperature is represented explicitly by the addition of new elements to the device model, and a thermal subcircuit to model heat flow. A new row and column was added to the Jacobian along with new state variables for temperature and power dissipation. The dc, ac and transient analyses can be all implemented using this thermal node approach. Detailed FORTRAN code modifications to SPICE2G.6 for this approach can be found in Appendix B.

The DC Solution

The following is a brief description of the changes made to the routine that computes the dc solution for the BJT. Starting where the derivation of (3.6) left off, the new matrix elements are derived. If the substitutions of collector thermal transconductance, $g_{TC} = I_C D_C$, and temperature,

$$T = R_{TH}P, \tag{3.31}$$

are made in (3.6) then I_C becomes

$$i_c = g_o v_{ce} + g_m v_{be} + g_\mu v_{cb} + g_{TC} T \quad . \tag{3.32}$$

A similar approach used for the base current results in the expression

$$i_b = g_\pi v_{be} - g_\mu v_{cb} + g_{TB} T \tag{3.33}$$

where $g_{TB} = I_B D_B$ is the base thermal transconductance. Using the electrical analogy for (3.31) results in modeling the temperature as a voltage and the power as a current. To include the power in the Jacobian, it must also be represented as a linear function of voltages and conductances. Substituting (3.32) and (3.33) into the expression for power in (3.2) gives the following:

$$\begin{aligned} p = {} & (g_o v_{ce} + g_m v_{be} + g_\mu v_{cb} + g_{TC} T) V_{CE} \\ & + (g_\pi v_{be} - g_\mu v_{cb} + g_{TB} T) V_{BE} + I_C v_{ce} + I_B v_{be} \end{aligned} \tag{3.34}$$

Using the above equations (3.31) through (3.34) the intrinsic BJT element stamp is formed:

$$\begin{bmatrix} g_m + g_o + g_\mu & -g_\mu & -g_m - g_o & g_{TC} \\ -g_\mu & g_\mu + g_\pi & -g_\pi & g_{TB} \\ -g_m - g_o & -g_\pi & (g_m + g_o + g_\pi) & (-g_{TC} - g_{TB}) \\ g_{pc} & g_{pb} & -g_{pc} - g_{pb} & R_{TH}^{-1} + g_{pt} \end{bmatrix} \begin{bmatrix} v_c \\ v_b \\ v_e \\ T \end{bmatrix} = \begin{bmatrix} -i_c \\ -i_b \\ i_c + i_b \\ P \end{bmatrix} \tag{3.35}$$

where $g_{pc} = v_{ce}(g_m + g_o + g_\mu)$ is the collector power term, $g_{pb} = v_{be}(g_\pi - g_\mu)$ is the base power term, and $g_{pt} = v_{be} g_{TB} + v_{ce} g_{TC}$ is the temperature power term.

The device temperature change is computed at the new device node by a controlled current source equivalent to the device power dissipation in series with the effective thermal impedance as shown in Figure 3-3. The ambient or global temperature of the circuit is represented by ground, and the T_A node in the figure is grounded for most cases.

The power dissipated in the transistor flows through the thermal impedance producing the effective temperature change of the device. The base and collector currents are then modified by the new thermal transconductance elements and the temperature. The thermal resistance is the sum of the resistances in the thermal impedance network, shown

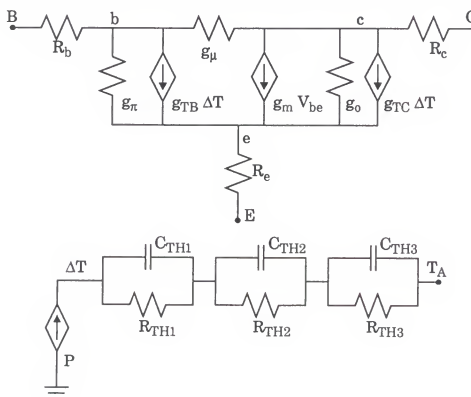

Figure 3-3 BJT internal large-signal circuit model showing new temperature node and thermal impedance ladder network.

here divided into three poles. Normally the thermal impedance network is connected to ground at the point T_A, but it can be coupled to other devices through an additional network representing the impedance between BJTs.

Implementation Options

Two choices affected the selection of an implementation strategy. The first choice was how to implement the thermal impedance subcircuit. The second choice was how to compute the temperature derivatives.

One way to implement the thermal subcircuit is similar to the implicit node approach above and would entail coding the subcircuit inside of SPICE. This requires the BJT matrix size to increase by a row and column for each new node in the impedance network. For maximum accuracy this could require as many as five new nodes, which would almost double the size of the BJT matrix. The topology of the impedance network would also have to be chosen in advance. The form of the ladder network shown in Figure 3-3 or the distributed form shown in Figure 3-4 could be used. Once implemented in the code, whichever topology is chosen cannot be changed by the user. The advantages of the internal approach are its

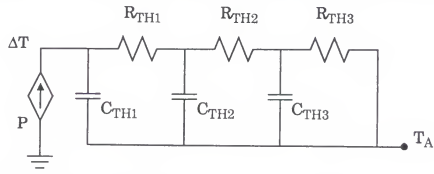

Figure 3-4 Distributed form of the thermal impedance subcircuit.

transparency to the user and the ability to use the bias-dependent dc and ac impedance model implemented in the implicit node approach.

The other way to implement the thermal subcircuit is to leave it to the user to specify the thermal impedance network in the circuit description file just like any other circuit elements. With this approach the flexibility to use any topography is retained. Computing the thermal impedance is a task well suited for pre-processing using the SPICE input processing routines. The biggest advantage to this approach is that relatively few changes are needed to the BJT matrix and SPICE data structures, making the implementation much simpler.

The temperature derivatives can be computed analytically or numerically. When this work began the main interest was in self-heating effects on dc and ac simulations in the forward-active region where the effects are greatest. An analytical solution of the temperature derivatives was practical when considering only the forward-active region, and made for an elegant implementation in the implicit node approach. However, a problem arose when a device passed through operating regions where the derivatives were incorrect. The simple problem of implementing a single region became complex when analytical expressions for other regions had to be implemented. A numerical approach to computing the temperature derivatives improves the problem of regional boundaries. The numerical derivatives are computed using

$$\frac{dx}{dT} \approx \frac{x' - x}{T' - T} \tag{3.36}$$

where x is any temperature dependent variable at the temperature T and x' is the value of that variable at temperature T' a small increment above T. To implement the numerical derivatives requires each variable, in this case the collector and base currents, to be computed twice, once at each temperature. This takes about the same amount of computation as the direct analytical approach because of the complexity of the analytical expressions (see equations (3.19)-(3.21)) compared to those of the currents.

SPICE Code Modification

This section details the implementation. While many different approaches have been tried over the course of this work, only the final version is covered here. In this implementation the best trade-offs for efficiency, flexibility, and convergence have been made. The thermal impedance is described externally in the SPICE input file, and the derivatives of the collector and base currents with respect to temperature are computed numerically.

Since the thermal impedance implemented externally to the BJT model, there are no new model parameters. Only one new device matrix node has been added for the temperature and four new state variables: temperature, power dissipation, and the derivatives of the collector and base currents with respect to temperature. The modifications to the SPICE data structure are relatively small, but are complicated to realize

because of the complexity of the SPICE data structure. Fifteen subroutines were modified and one new routine added to incorporate the changes needed for self-heating, but not all of these are discussed here for the sake of brevity. For more details, see Appendix B.

The format of the BJT device input line was changed to include the external thermal node. The new BJT line is as follows:

Qxxxx nc nb ne [ns [nt]] model [area] [initial conditions]

where the items in brackets are optional. The substrate node number (ns) and the temperature node number (nt) are both optional. The substrate node must always be given if the temperature node is used or differentiating between them when only one is specified would be impossible. An example circuit description with self-heating is shown in Figure 3-5 where the substrate node has been grounded. If the thermal node is not specified the node is assumed grounded and the standard isothermal model is used.

Once the preprocessing routines were modified to recognize the thermal node, build the new device matrix row and column, and add the new state variables, the work of modifying the device model routine for dc and transient analyses could begin. Note that the temperature state

```
VCE 1 0 DC 5
VBE 2 0 DC 0.8
Q1 1 2 0 0 3 NPNMODEL
RTH 3 0 500
```

Figure 3-5 Sample modified BJT circuit description for SPICE with a thermal resistance of 500 K/W.

variable stores the change in temperature above the circuit ambient, and is converted to the absolute temperature only where needed in the model. Unless otherwise stated, the temperature referred to in the following is the temperature change.

In the device initialization phase, the temperature is set to zero. No power has been computed so no temperature change can exist. Next in the linear extrapolation section the equation for predicting the power is added and temperature is added to the current prediction equations:

$$\hat{P} = P + |I_C \Delta V_{CE}| + |I_B \Delta V_{BE}| \tag{3.37}$$

where $\hat{P}$ is the predicted power and P is the power from the previous iteration,

$$\hat{I_C} = I_C + (g_m + g_o)\Delta V_{BE} - (g_o + g_m)\Delta V_{BC} + g_{TC}\Delta T, \tag{3.38}$$

and

$$\hat{I_B} = I_B + g_\pi \Delta V_{BE} - g_\mu \Delta V_{BC} + g_{TB}\Delta T. \tag{3.39}$$

Next the initial convergence check is changed to include the temperature. The absolute tolerance on the temperature multiplies the conversion factor of 100 K/V and the tolerance of the junction voltages. With only a unity conversion factor, using the default voltage tolerances would require the temperature to be accurate to 50 μK for convergence. This is more precision than is needed and discourages convergence. Scaling the tolerance to 5 mK provides enough accuracy and improves the robustness of convergence. The power does not need to be included in the

check, because it is a linear function of currents and voltages and will converge when they do.

Similar to the implicit node approach, the temperature update equations were moved inside the model routine. The model parameters are adjusted for local changes in temperature in each iteration. After the initial convergence check and any junction voltage step-size limiting, the temperature dependences are inserted. A loop is formed around the temperature-update equations, so they can be used in computing the numerical derivatives of the currents with respect to temperature. (See the flowchart in Figure 3-6.) The temperature in augmented by 0.1 K, the model parameters are adjusted for temperature, and the currents are computed from the model equations. At this point the high-temperature current values are temporarily stored and the 0.1 K is subtracted back from the temperature to return to the original value. The loop returns to the temperature update and then recomputes the currents for the proper temperature. The loop is exited and the current derivatives, g_{TC} and g_{TB}, are computed using the approximation of (3.36). The choice of 0.1 K for the temperature change was arrived at empirically; it results in enough change in the currents so the numerical problem of subtracting two close numbers of finite precision does not occur, and it is small enough to preserve accuracy in the approximation.

The power dissipation must be computed before any changes from reactive currents in the transient analysis can change the base and

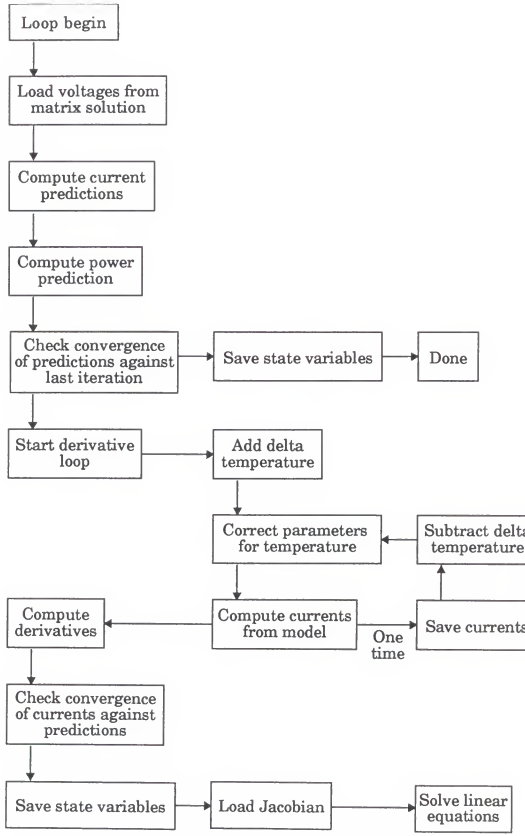

Figure 3-6 Flow chart of BJT model routine.

collector current value. The changes in current from reactive elements in the transient analysis are not real and do not contribute to real power dissipation. Such charging currents must be dissipated in the device parasitic resistances to generate temperature changes. Since the reactive elements of the thermal impedance are defined externally, no changes are needed to the transient analysis section of the routine.

SPICE includes a second convergence check after the transient section to compare the currents newly computed from the model to the predicted currents. As noted previously, it is not necessary to check the power; the temperature will not change within the model routine. The state variables are saved and the Jacobian loaded as shown in (3.34) and (3.35).

The changes to the ac analysis routine are minimal. In addition to the normal matrix load, the temperature row and column are loaded with the small-signal equivalent of the dc matrix load.

Advantages and Limitations

Many of the advantages and limitations to the external thermal impedance approach have already been mentioned, but is worth restating them. The advantages are many. Since the thermal impedance is implemented outside the BJT model, any topology or number of time constants can be used for the thermal equivalent circuit. No new model parameters and only a few new state variables are required; an internal thermal impedance would have many of each. The temperature

derivatives are computed numerically, improving continuity across operating region boundaries with little or no additional computational overhead. Transient analysis is possible where it was not in the implicit node approach. The biggest limitation is that the thermal impedance is fixed. Neither bias nor temperature dependences are implemented, and the smooth frequency-domain solution of (2.20) is unworkable in the small-signal analysis. Versions of SPICE allowing nonlinear equations in user-defined controlled sources would make it possible to overcome this limitation.

Convergence

Self-heating makes convergence harder to reach. The most common problem with convergence is thermal runaway. SPICE sets the junction voltages to large values on the first iteration to ensure the devices are on. With the large forward biases the model equations compute large currents. After the first matrix solution, the returned node voltages tend to be quite large. These factors combine to over-predict the power dissipated in the device, causing an over-prediction of the temperature. The over-predicted temperature in the second iteration can cause the temperature-dependent parameters to become large and cause the computed currents to increase enough to cause the temperature to increase even more. The temperature and current will grow geometrically until a numerical overflow occurs, usually after only a few iterations.

Sometimes a variation of this thermal runaway occurs where the simulated temperature becomes high in an iteration and the model overcompensates causing a negative temperature change, then positive again, oscillating with increasing magnitude with similar results as before.

Two controls have been put in place to minimize these problems: a hard limit that ensures that only positive temperature changes and a method in which the initial temperature-iteration is skipped. Theses enhancements were made to both implementations to improve the likelihood of convergence by preventing the thermal runaway condition.

For the iteration-skipping method, the thermal resistance is always held to zero on the first iteration. This is done by checking the initialization flag and skipping the power computations when the flag is set. The zero power from the first iteration results in zero temperature change for the second iteration. The BJT currents and voltages usually settle into the solution region by this time, and the inclusion of self-heating is a much smaller perturbation on the solution. The resulting temperature computed from the power at this point will be closer to the actual value for the BJT, and subsequent iterations usually converge. The BJT model effectively executes two isothermal iterations before the self-heating is turned on.

To stop the oscillation a hard temperature limit was added at zero degrees local temperature change that prevents negative temperature

changes. If the model computes a negative temperature change at the beginning of any iteration the temperature change is set to zero for the remainder of that iteration. This allows the model to recover from the oscillation before it becomes a problem, and it prevents endothermic solutions not supported by the Gummel-Poon model.

A third method could be used to constrain runaway conditions in nonlinear equation solutions: step-size limiting. This method limits the amount a solution element (for example: V_{BE}) can change in a single iteration. This method can work for temperature in the self-heating model, but the best value to use for the maximum step size is difficult to determine and varies greatly from device to device and circuit to circuit. A step size that is too small can slow down convergence noticeably and one that is too large may only slow down runaway but not prevent it. For the BJT model this approach for convergence control gives little return for the computational investment compared to the other two implemented methods, so it is not used in this implementation. It could be useful, however, for special cases, especially if the user had control of the temperature step-size for each transistor.

Results of convergence comparisons between the original isothermal model and the explicit thermal node approach are shown in the next chapter. Such comparisons provide only general estimates of the costs of using self-heating, since convergence in SPICE has seemingly random behavior for large circuits. For example, modifications to the input such as

changing the order in which the elements are listed can affect convergence.

CHAPTER 4
SELF-HEATING IN CIRCUITS

Self-Heating Effects

An easy way to understand self-heating effects in circuits is through the effects on BJT small-signal parameters [Mül64, Mül70]. Assume that with self-heating neglected the common-emitter g-parameters are given by

$$g_{11E} = 1/r_{\pi E} \qquad (4.1) \qquad g_{12E} \cong 0 \qquad (4.2)$$

$$g_{21E} = g_{mE} \qquad (4.3) \qquad g_{22E} = 1/r_{oE} \qquad (4.4)$$

Using the partial derivatives in Chapter 3, when self-heating is considered, the small-signal collector current becomes

$$i_c = \frac{g_{21E}v_{be} + g_{22E}v_{ce} + D_C R_{TH} I_C (I_B v_{be} + I_C v_{ce})}{1 - D_C R_{TH} P} \qquad (4.5)$$

assuming that $i_c/i_b \cong I_C/I_B$. Similarly, the base current is

$$i_b = \frac{g_{11E}v_{be} + g_{12E}v_{ce} + D_B R_{TH} I_B (I_B v_{be} + I_C v_{ce})}{1 - D_B R_{TH} P} \qquad (4.6)$$

Thus the g-parameters, corrected for self-heating, can be expressed as

$$g_{11} = \frac{g_{11E} + D_B R_{TH} I_B^2}{1 - D_B R_{TH} P} \qquad (4.7) \qquad g_{12} = \frac{g_{12E} + D_B R_{TH} I_B I_C}{1 - D_B R_{TH} P} \qquad (4.8)$$

$$g_{21} = \frac{g_{21E} + D_C R_{TH} I_B I_C}{1 - D_C R_{TH} P} \qquad (4.9)$$

$$g_{22} = \frac{g_{22E} + D_C R_{TH} I_C^2}{1 - D_C R_{TH} P} \qquad (4.10)$$

The denominators in (4.7) - (4.10) become significant as the power approaches the critical values $P_C = 1/D_C R_{TH}$ or $P_B = 1/D_B R_{TH}$ where plots of I_C and I_B versus the port voltages have infinite slopes. These limits are related to the onset of second breakdown. In most analog circuits the power dissipation is kept much less than P_C or P_B, so that the effects of the denominator are negligible except in circuits requiring great precision. Note that as values for R_{TH} rise, the power required to cause errors decreases.

Now consider g_{21}. The electrical-only transconductance g_{21E} is approximately $g_{mE} = I_C/V_T$. If the denominator in (4.9) is close to unity, then the value of g_{21}, corrected for self-heating, is approximately $g_{21E}(1 + D_B R_{TH} I_B V_T)$. Similarly, the electrical-only input conductance is $g_{11E} = 1/r_{\pi E} = I_B/V_T$, so that $g_{11} = 1/r_\pi \cong g_{11E}(1 + D_B R_{TH} I_B V_T)$. For any reasonable conditions, the term $D_B R_{TH} I_B V_T$ is very small, so these two parameters are only affected by the corrections in the denominators, which require substantial power dissipation approaching P_B.

The situation is different for the other two g-parameters. A simple model for the electrical output conductance is $g_{22E} = 1/r_{oE} = I_C/V_A$, where V_A is the Early voltage. From (4.10), for $P \ll P_C$, $g_{22} = g_{22E}(1 + D_C R_{TH} I_C V_A)$, where the second term in the parentheses can be thought of as a figure of merit giving the fractional error in the output

conductance caused by self-heating for low power. This term can be substantial even for very modest currents.

Neglecting self-heating, the effects of changes in V_{CE} on I_B, and thus g_{12E}, are very small in forward-active operation; typically g_{12E} can be neglected in circuit analysis. However, including self-heating and assuming the denominator in (4.8) to be near unity, this parameter is much larger in magnitude and positive:

$$g_{12} \cong D_B R_{TH} I_B I_C = g_{22E} D_B R_{TH} I_C V_A / \beta_0 \tag{4.11}$$

This effect is exploited in the next chapter to extract the thermal impedance of the BJT.

Simulation Examples

The repercussions of the above results for circuit designers are evident in a variety of circuit types. The following examines the effects of self-heating on some common circuits. The SPICE circuit descriptions for all the circuits shown are provided in Appendix D. All the simulations used the explicit thermal node simulator with thermal impedance subcircuits.

The most common analog circuit is the common-emitter amplifier configuration where the emitter and base impedances are typically small and output (collector) load is large, comparable to r_o. Here self-heating can significantly raise the input resistance and decrease the output resistance. The result is a greatly reduced gain at low to moderate frequencies where self-heating predominates. This effect is illustrated in Figure 4-1

[Fox93b]. This common-emitter amplifier circuit was originally published by Lee using the implicit temperature simulator developed in this work [Lee92]. The rest of the circuits are new examples of self-heating effects on circuits.

The larger 741 op-amp circuit shown in Figure 4-2 is a common two-stage amplifier circuit with a low current, low gain differential input stage and a high current, high gain stage [Gra84]. The dc transfer

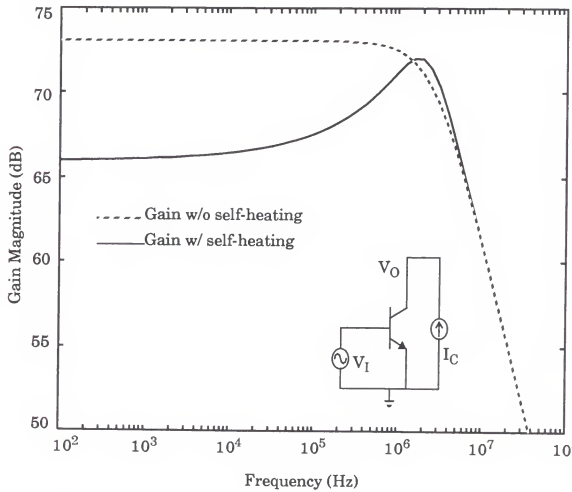

Figure 4-1 Simulated voltage gain vs. frequency for common-emitter amplifier with and without self-heating. Note that the dc gain is halved by self-heating.

characteristics of a 741 op-amp circuit are shown in Figure 4-3 both with and without self-heating. The self-heating curve shows a significant nonlinear distortion. This distortion causes a 10% shift in offset voltage and a large reduction in gain. The greater distortion at low output voltage

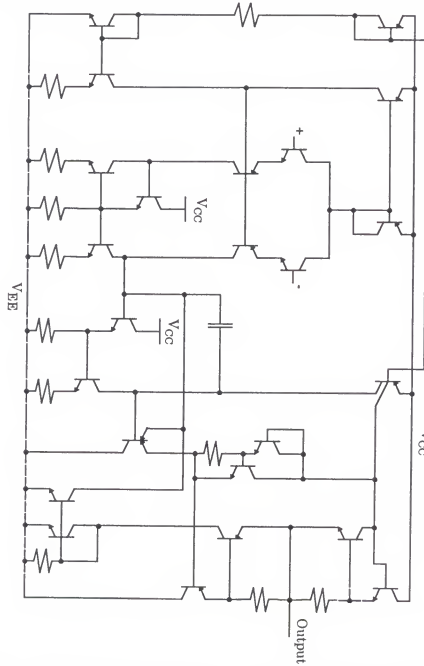

Figure 4-2 Schematic for the 741 operational amplifier circuit. Resistance values are as given by Grey and Meyer [Gra84]

indicates the PNP current-mirror in the second amplifying stage, which has no emitter degeneration, has a much reduced output resistance due to self-heating.

For the 741 op-amp, the first stage is operated at such a low current that self-heating effects are minimal. The second stage is operated at higher currents, and is driven from the low output impedance of an emitter follower, so there is an error in the dc gain due to self-heating. This is shown in Figure 4-4. When Miller compensation is applied around the

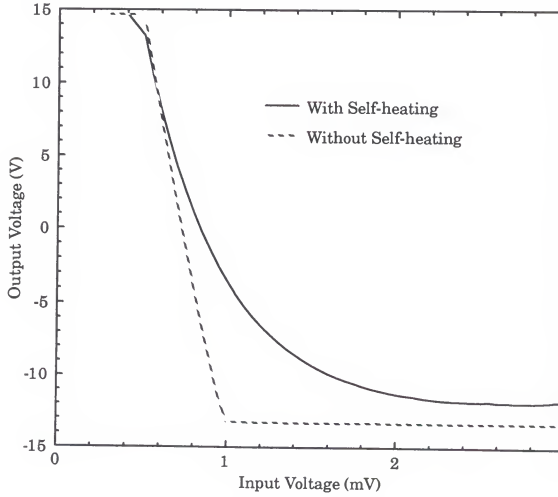

Figure 4-3 Simulation of dc transfer characteristics of a 741 op-amp with and without self-heating.

second stage, the dominant pole frequency is also shifted by self-heating. However, the negative feedback through the compensation capacitor swamps out the positive thermal feedback above the dominant pole frequency, so there is little thermal effect on the phase margin.

Some large-signal circuits where precise V_{BE} matching is important can be affected by self-heating when the collector-emitter voltages of the matched BJTs differ. The BJT with the higher V_{CE} has greater power

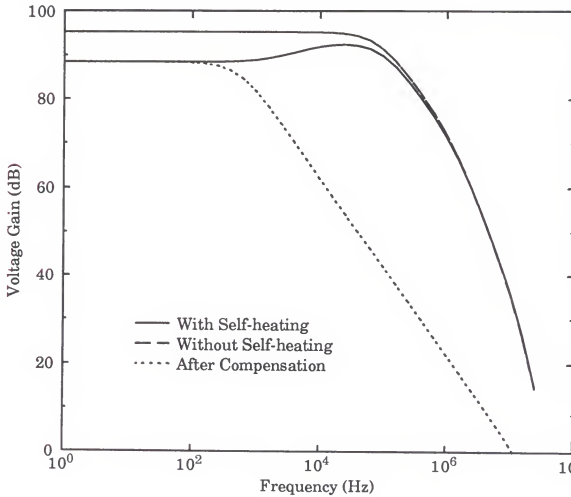

Figure 4-4 Simulation of a 741 op-amp open-loop voltage gain with and without self-heating showing the effects of compensation.

dissipation, causing its V_{BE} to decrease to suppress the increase in collector current. These dc effects were demonstrated for current mirrors by Munro and Ye [Mun91], and for current reference circuits by Fox and Lee [Fox93b]. When these circuits are used in switching applications the V_{BE} shifts settle over long time periods as the BJTs heat and cool.

Consider the pin-driver circuit shown in Figure 4-5. The function of this circuit is to transfer a voltage level at low-current, high impedance circuit to a high-current, low-impedance circuit (or package pin). Going from V_{IN} to V_{OUT} the signal increases two diode drops through Q_1 and Q_2

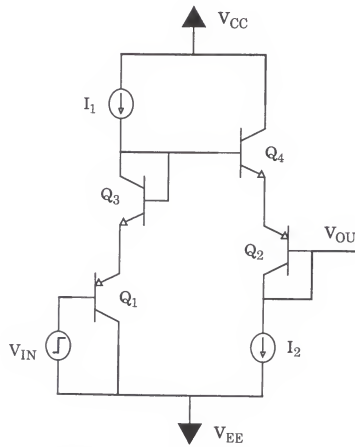

Figure 4-5 Simplified pin-driver circuit.

then decreases two diode drops through Q_4 and Q_2. If all of the transistors are matched the output follows the input.

Transistors Q_3 and Q_2 are diode-connected and do not dissipate much power. Q_1 and Q_4 can have large collector-base voltages and can dissipate significant power. Figure 4-6 shows the temperature transients of Q_1 and Q_4. For this simulation, the supply voltages V_{CC} and V_{EE} were 10 V and -10 V, the current sources I_1 and I_2 were 1 mA and 10 mA, and

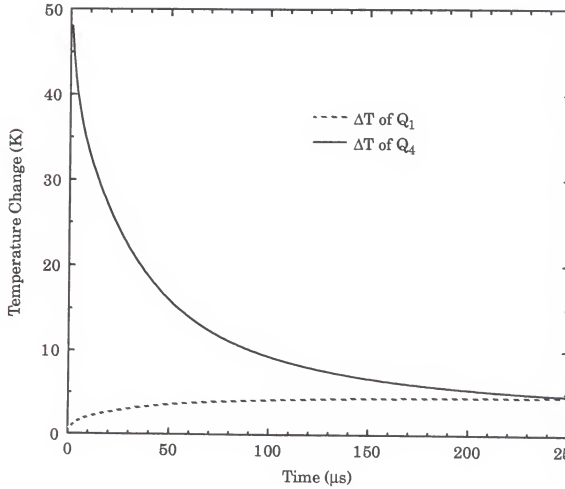

Figure 4-6 Simulated temperature transients of Q_1 and Q_4 of the pin-driver circuit for and 18 V input pulse.

the input voltage pulse switched from 2 V to 18 V. The temperature swing of Q_4 is much larger than that of Q_1.

The output voltage transients are shown in Figure 4-7 with and without self-heating. The long thermal transient can be seen in the long output voltage settling time. While the 0.1% settling time for the isothermal case is less than 10 ns, with self-heating the settling time is about 100 μs. The long thermal tail is a problem for high-speed precision applications.

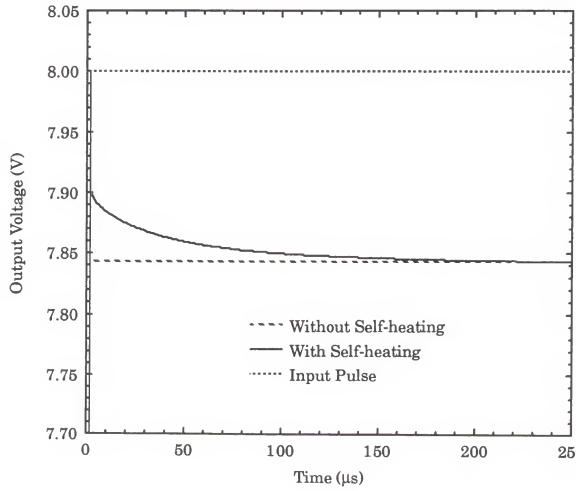

Figure 4-7 Simulated transient output voltage for the pin-driver circuit with and without self-heating.

Another application of self-heating in circuit design is thermal coupling of transistors that are near each other. The distance for a device to be considered near to another in thermal terms depends on the circuit. Fukahori and Grey showed that for a operational amplifier the input transistors could be affected by the heat generated in high-current output transistors if they are located close to the input transistors [Fuk76]. They showed that careful layout that located the output stage away from the input stage could cure this problem. The general rule in these cases is to lay out high-power elements of the circuit away from the sensitive subcircuits of a design.

For the case of multiple-emitter BJTs the above rule does not apply. These devices are used in high-frequency applications primarily because of the reduced parasitics from the shorter intrinsic base region and reduced current crowding of the thinner emitter fingers. To simulate a multi-emitter BJT with self-heating, it can be separated into multiple devices, one for each finger. This method was developed in collaboration with Brodsky [Bro97b]. A schematic for a three-finger BJT is shown in Figure 4-8. The fingers are thermally coupled by current-controlled voltage sources (H-sources in SPICE). Each finger's thermal subcircuit includes an H-source for each of the other fingers. An H-source is controlled by the power flowing in another thermal subcircuit multiplied by the thermal resistance between the two fingers.

Figure 4-9 shows the result of a dc simulation of the three-finger BJT. The middle finger is heated more than its neighbors because of its close proximity to both of the side fingers. It carries more of the base current and contributes more to the total collector current. The result is a reduced output impedance over the non-thermal simulation of about 70% for this example.

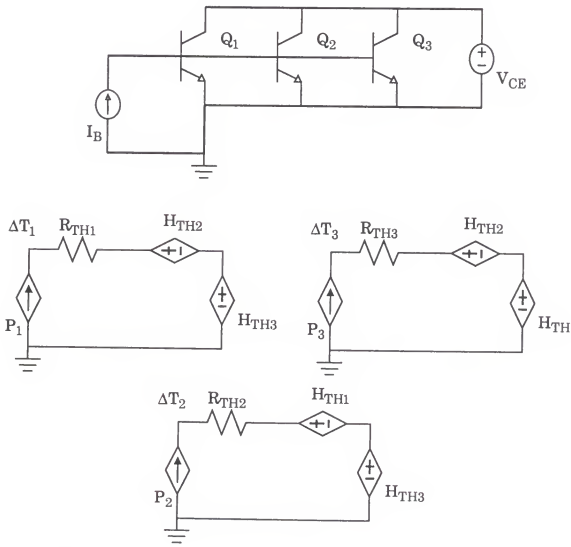

Figure 4-8 Schematic for simulating a three-finger-emitter BJT, and the three coupled thermal subcircuits.

The simulation of circuits with self-heating adds to the accuracy of many designs, but is not necessary for all circuits. To further increase the reliability of simulation results, more accurate model parameters are needed, including thermal impedance parameters. A technique for extracting the thermal impedance for BJTs is introduced in the next chapter. The effects of self-heating on the Gummel-Poon parameter extraction are also examined.

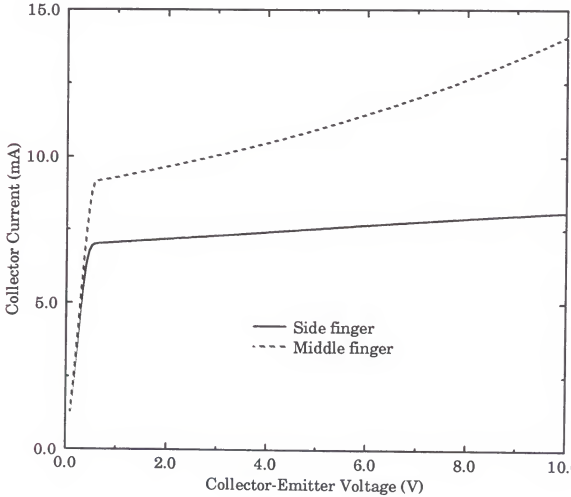

Figure 4-9 Contribution of the middle and side fingers to the total collector current of a three-finger emitter BJT.

Convergence Comparison

The simulation statistics for each of the above circuits were collected for the self-heating and isothermal cases. The results of a three run average are shown in Table 4-1. The statistics for the multi-finger

Table 4-1 Circuit simulation statistics.

Circuit	Operating-point Normal/Self-heating number/time (s)		Ac Normal/Self-heating number/time (s)		Transient Normal/Self-heating number/time (s)	
Common-emitter NPN	5/0.03	6/0.03	46/0.07	46/0.12		
741 Op-amp	15/0.45	15/0.95	46/1.08	46/1.92		
Pin-driver	8/0.07	9/0.12			710/2.70	856/5.85

transistor are omitted, because there is not an isothermal equivalent circuit for comparison. These statistics are reported by the ACCT option in SPICE the reports on execution times and iterations for each analysis. The circuits were simulated on a Sun Microsystems, SPARCstation 10 with 64 MB of memory. The computer, at the time of simulation, was only running the FVWM window manager and common UNIX daemons, so the simulation time for each circuit was relatively unaffected by resource demands by other processes. The quality of the computer isolation was shown to be effective, since each of the averaged simulations returned the same result.

It is clear that simulations using self-heating have a greater cost in computation time. For the case of the single-BJT common-emitter circuit difference in the operating-point solution times is lost in the accuracy of

the measurement, but the self-heating circuit needed an extra iteration implying the need for greater computation time. The ac analysis used 71 percent more time because of the added elements of the thermal impedance subcircuit. Because of the linear nature of the ac analysis, the number of iterations is always exactly the same as the number frequency points computed.

For the 741 op-amp, the iteration count for the operating-point solution is surprisingly the same, but there is a 111 percent increase in computation time for self-heating. It is important not to infer too much from these results, because the bias used for each case was different due to the shift in operating-point caused by self-heating. A small bias change for this class of circuits will almost always have an effect on convergence independent of self-heating. The additional time is due to not only the added circuit elements but the increase computation need in the BJT model for self-heating. The ac solution took 78 percent longer, which is consistent with the common-emitter circuit results.

The pin-driver circuit's initial operating-point solution required one more iteration and 71 percent greater time per iteration for self-heating. The effects of self-heating on transient iterations and computation time are expressed in the time-step control. The SPICE algorithm to compute the time-step in the transient analysis increases the size of the time step each iteration until convergence is not reached and then "backs up" to try a smaller time-step. For the isothermal simulation, there are practically

no changes in the output for the majority of the analysis time. SPICE can take large time-steps and complete the simulation with fewer iterations. For the self-heating case the long thermal tail requires smaller time-steps and a greater number of iterations. The time per iteration increased 80 percent.

To keep the added overhead of self-heating to a minimum, it is important to use the insight provided in the first part of this chapter to choose the devices most affected for simulating with self-heating. This mixed approach also can improve convergence behavior over using self-heating for every BJT in the circuit.

This chapter has used models for thermal impedance for simulation. In the next chapter, the technique used for extraction of the thermal impedance is presented. It was shown here that self-heating changes device performance. The next chapter shows how self-heating can affect the extraction of the Gummel-Poon model parameters.

CHAPTER 5
MEASUREMENT OF MODEL PARAMETERS

This chapter describes methods for extracting model parameters for use in self-heating circuit simulation. First, a two-part, dc and ac method for extracting the thermal spreading impedance is shown. The results can be used to either verify an impedance model or to directly extract the impedance network for each device [Zwe95a, Zwe96]. Second, the effects of self-heating on Gummel-Poon dc model parameter extraction are examined. Self-heating causes errors in certain high-current parameters if they are extracted under nominally isothermal conditions unless self-heating is accounted for in the optimization phase [Dav88].

Measurement of the Thermal Impedance

Two components must be characterized for an accurate model: the thermal spreading impedance (temperature response to a change in dissipated power), and the temperature dependence of the current-voltage relationships of the transistor. At first glance measuring the temperature dependence of the currents appears straightforward; the ambient temperature is varied by a known amount and the resulting changes in the terminal currents are measured. Actually the situation is more complicated because the actual device temperature rises above the ambient by an unknown amount as a result of self-heating. This unknown

temperature change can be significant compared to the ambient temperature change, especially in high-current regions. This chapter describes a way to overcome this problem, and a way to use the results to extract the thermal impedance from transient response.

Measurement of the base current response is treated as the thermometer in the techniques described here. An underlying assumption of these measurements is that collector-base conductance from the modulation of the base recombination current is negligible compared to the effect of self-heating on the total base current. If operation is in the forward-active region where significant impact-ionization can be avoided, any change in base current resulting from a change in collector-base voltage with constant base-emitter voltage is dominated by self-heating. Characterization of the thermal impedance requires two steps. The first step involves finding the fractional temperature coefficients: the fractional change in base or collector current for a unit change in the transistor temperature with fixed base-emitter voltage. We find this information by sweeping the dc collector voltage of the transistor at different ambient temperatures and then cancelling the self-heating. The second step is to find the thermal impedance from the base current response to a collector voltage step. Using the temperature coefficient results from the first part, the base-current response is converted to a temperature response. After normalizing the transient temperature response by the dissipated power, the thermal impedance model can be fitted to the waveform.

Temperature Coefficient Measurement

In this part of the measurement cycle the goal is to calibrate the base-current thermometer by measuring the change in current caused by a change in temperature with fixed base voltage. When the temperature is increased the collector current and the power increase, causing the local temperature to increase considerably because of self-heating. An error arises whenever self-heating contributes significantly to the device temperature (at moderate to high current levels). Transient measurements, to be discussed in the next section, are typically made in these regions, and there needs to be a way to characterize the base current dependence on temperature in the presence of self-heating. The total temperature change of a BJT, including self-heating, is given as

$$\Delta T = R_{TH}\Delta P + \Delta T_A = R_{TH}(I_C\Delta V_C + \Delta I_C V_C) + \Delta T_A \qquad (5.1)$$

where ΔT is the total temperature change at the emitter, ΔT_A is the ambient temperature change, and R_{TH} is the thermal resistance. Separately measuring the change in base current caused by a change in ambient temperature and then by a change in power gives two equations that allow cancellation of the self-heating from the temperature coefficients. The power is controlled by varying the collector voltage in the forward-active region where base-collector conductance is negligible and avoiding regions where impact ionization is significant.

When changing the ambient temperature with the collector voltage, V_C, constant, (5.1) gives a net temperature change of

$$\Delta T\big|_{\Delta V_C = 0} \cong \Delta T_A + R_{TH} V_C \Delta I_C \tag{5.2}$$

where

$$\Delta I_C = \frac{\partial I_C}{\partial T}\Delta T = D_C I_C \Delta T \tag{5.3}$$

is the change in collector current due to a change in temperature. Substituting into (5.2) and solving for ΔT gives

$$\Delta T\big|_{\Delta V_C = 0} = \frac{\Delta T_A}{1 - D_C R_{TH} I_C V_C}. \tag{5.4}$$

When T_A is held constant and V_C is varied, (5.1) reduces to

$$\Delta T\big|_{\Delta T_A = 0} = R_{TH}(I_C \Delta V_C + \Delta I_C V_C) \tag{5.5}$$

where

$$\Delta I_C = \frac{\partial I_C}{\partial T}\Delta T + \frac{\partial I_C}{\partial V_C}\Delta V_C = D_C I_C \Delta T + \frac{\Delta V_C}{V_A} I_C \tag{5.6}$$

and V_A is the Early voltage. Substituting into (5.5) and solving for ΔT gives

$$\Delta T\big|_{\Delta T_A = 0} = \frac{I_C R_{TH}\left(1 + \dfrac{V_C}{V_A}\right)\Delta V_C}{1 - D_C R_{TH} I_C V_C}. \tag{5.7}$$

Notice that the denominators of (5.4) and (5.7) are the same. Combining (3), (5) and the base-current dependence given by

$$\Delta I_B = \frac{\partial I_B}{\partial T}\Delta T + \frac{\partial I_B}{\partial V_C}\Delta V_C \approx D_B I_B(\Delta T_A + R_{TH} V_C \Delta I_C) \tag{5.8}$$

where the second term is the base-collector conductance (assumed to be negligible), results in the following ratio:

$$\frac{(\Delta I_B/\Delta V_C)\big|_{\Delta T_A = 0}}{(\Delta I_B/\Delta T_A)\big|_{\Delta V_C = 0}} = I_C R_{TH}(1 + V_C/V_A). \tag{5.9}$$

This can be solved for R_{TH} if V_A is known or is much greater than V_C. It is important to note that the thermal resistance found here is an effective value, and includes the thermal impedance of the package and die. While this is not an exact value for the intrinsic thermal resistance, it is a good estimate for extremely low frequencies.

Once R_{TH} is known, (5.8) can be solved for the fractional temperature coefficient of base current:

$$D_B = \left.\frac{\Delta I_B/I_B}{\Delta T_A + R_{TH} V_C \Delta I_C}\right|_{\Delta V_C = 0}. \tag{5.10}$$

Note that this rather complicated procedure is not needed at moderate to low currents, where self-heating does not need to be cancelled.

The following is a description of the measurement setup and algorithm. The transistor is connected in common-emitter configuration inside an oven or on a probe station thermal chuck. It is important to have stable, accurate ambient temperature control during the measurement, because the voltage sweep takes a long time and errors in the ambient temperature significantly affect the accuracy of the results.

The algorithm for the measurement is simple, though very time-consuming. First, set the ambient temperature and allow it to settle completely, chose a base voltage for the region of interest (moderate to

high current region), and sweep the collector voltage in the forward-active region where a noticeable, linear increase in I_B can be seen (avoiding any fall-off due to impact-ionization). Use a very long delay time between measured points; about 30-60 s typically works well. This allows the self-heating of the transistor to settle before making the measurement at each point. Execute the sweep, measuring I_C, I_B, and V_{CE} for each V_{BE} of interest. Change the temperature and repeat the voltage sweeps done before (from experience, a temperature step of less than 10 °C is recommended to avoid errors caused by the strongly nonlinear dependence of current on temperature). Take the average current changes with respect to collector voltage and ambient temperature. Using a geometric mean helps compensate for the non-linearity of the data. Use (5.9) and (5.10) above to compute the fractional temperature coefficients at each base voltage value.

Figure 5-1 shows a plot of the temperature coefficients versus base voltage for an NPN Harris UHF transistor with a 3x50 μm^2 emitter. The UHF process yields high-frequency, dielectrically isolated, vertical transistors [Dav92]. The collector voltage was swept from 2-4 V while the base voltage was varied from 700-850 mV in 5 mV intervals using a Hewlet-Packard 4145A Semiconductor Parameter Analyzer. External computer control was used to run the HP 4145A, because only short durations are available for the voltage steps using the integrated controller. The automation of the experiment also helped reduce human

errors and total measurement time by integrating setup, measurement, and data collection into a single routine. The ambient temperature was controlled using a Delta Design Model 111 Thermal Chamber; temperatures used were 25 and 33 °C.

This measurement has errors that are dominated by oven-temperature accuracy, non-linearity of the data, and effects of base and emitter resistances. The thermal chamber has a published repeatability

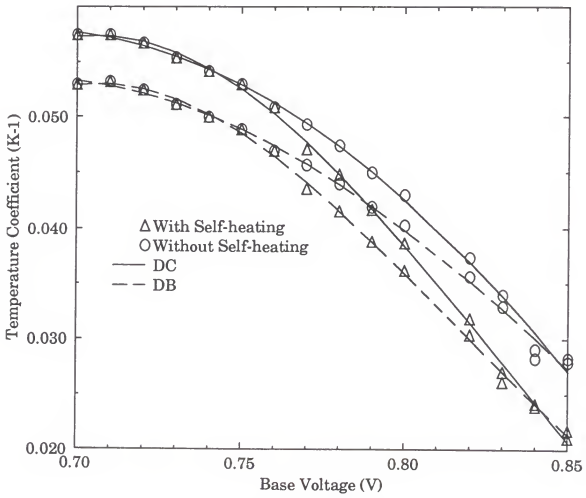

Figure 5-1 Harris 3x50μm UHFN3 bipolar transistor fractional temperature coefficients of base and collector currents versus base-emitter voltage measured with and without correcting for self-heating.

accuracy of 0.5 °C and a relative error of 0.1 °C. The absolute error is not as important for the temperature coefficient measurement as the relative error since the change in temperature is used in the computations. Because the measurements shown here were repeatable with an error consistent with the 0.1 °C, the published errors can be believed. The emitter and base resistances cause a slight error in the intrinsic V_{BE} for high currents. When ΔI_C is 1 mA the intrinsic ΔV_{BE} is reduced by 2 mV. The total error for the experiment is approximately 10 percent.

Thermal Impedance Measurement

The objective of this part of the measurement cycle is to measure the normalized thermal response to a step in power. This is done by converting the base current response into a temperature response using the temperature coefficients extracted above. Solving the fractional temperature coefficient definition for the change in temperature gives

$$\Delta T(t) = \frac{I_B(t) - I_B(0)}{I_B D_B} \tag{5.11}$$

where I_B is the median value of the base current for the transient. At this point the signal represents the thermal transient for the power step applied, and it must be normalized for a unit power step. This is done by dividing the $\Delta T(t)$ signal by the power of the step:

$$\Delta T_{norm}(t) = \frac{\Delta T(t)}{\Delta I_C V_C + \Delta V_C I_C} \tag{5.12}$$

where the denominator term is the magnitude of the power step.

The setup for this measurement is similar to the temperature coefficient common-emitter configuration with a collector voltage step applied instead of a slow voltage sweep. The setup is shown in Figure 5-2, where the BJT in the center is the test device being probed on a wafer. The current-to-voltage converter transforms the base current into a voltage waveform for the oscilloscope. With the base voltage held constant, the base current is

$$I_B(t) = (V_{OUT} - V_{BE})/R_F. \tag{5.13}$$

where R_F is the feedback resistance. Unfortunately, the intrinsic base-emitter voltage is not constant because of the parasitic base and emitter resistances. Since the change is small compared to the thermal voltage, a linear correction is suitable:

$$I_B(t) = \frac{V_{OUT} - V_{BE}}{R_F - R_B - (1+\beta)R_E} \tag{5.14}$$

where R_B and R_E are estimated base and emitter resistances and β is the large-signal current gain.

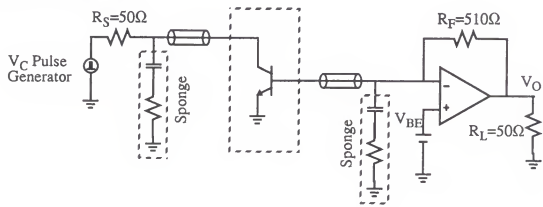

Figure 5-2 Schematic for the transient response measurement.

There are important practical trade-offs in choosing the feedback resistor value. The most important is the gain versus bandwidth trade-off. Gain must be large because the signals of interest are typically very small, yet a wide bandwidth of about 100 MHz must maintained to resolve the earliest part of the transient response. In our measurements, an amplifier with gain-bandwidth of 1 GHz was used with R_F = 510 Ω and R_{SCOPE} = 50Ω for a loop gain of about 10 and a bandwidth of 100 MHz. The "sponges" shown in Figure 5-2 are high-frequency sinks that absorb the energy at frequencies beyond those of interest in the thermal measurement; they are required to keep the UHF transistors from oscillating. The sponges also help absorb some of reflected power from the unmatched impedances of the pulse generator and BJT collector. The device under test was probed on-wafer using one meter, 50 Ω co-axial cables.

The following is a description of the method used to make the step-response measurement. First, a value of base voltage was selected to give a current large enough that the collector voltage step created enough power to get a measurable change in base current for a square wave collector voltage of 2-5 V. The best frequency to use for the square wave depends on the transistor; the response must settle before the end of each half cycle. Much of the response occurs in the first 50 μs, but it continues until about 1 ms, after which package heating dominates. Because of the distributed nature of the response, the op-amp output was recorded using three overlapping time windows of increasing size to capture with high

resolution the fast beginning of the response while keeping the number of data points to a minimum for the long tail. Next, the pulse generator was set to dc output and an ammeter was connected in series with the collector to measure the collector current at the pulse maximum and minimum values. The op-amp output is also measured at this time to compute the base current from (5.14). From this data the β of the transistor is determined, so that the transient base current can be converted into a collector current for computation of the power step magnitude. The base current minimum and maximum in this dc measurement differ from the values in the transient because of the difference in mean power being dissipated. Using (5.11) and (5.12), the normalized temperature response can be computed. Figure 5-3 shows the normalized temperature responses for a range of geometries.

Figure 5-4 shows the thermal resistance of the same transistors versus emitter width. Estimated errors for the measurement are also shown. These errors are primarily from three sources: errors in the extracted fractional temperature coefficient, the variance of β over the range of collector voltage, and the absolute error of the measured base current.

The absolute error of the base current seriously limits accuracy for small transistors that operate at low currents where it is difficult to get a large enough change in I_B to be seen over the noise of the measurement equipment. This does not mean that thermal effects in these transistors

are insignificant. In the setup used here the limit for dc collector current is approximately 1 mA. For a small transistor biased at 1 mA undergoing a 5 mW power step with an R_{TH} of 500 K/W, an approximately 15% increase in I_B occurs. However, this change is seen as only 1 mV at the op-amp output. This signal is hard to resolve with a oscilloscope with a 1 mV/division minimum resolution. Add to that a typical 2 mV of noise superimposed on this signal, and it becomes very difficult to resolve the transient even with averaging.

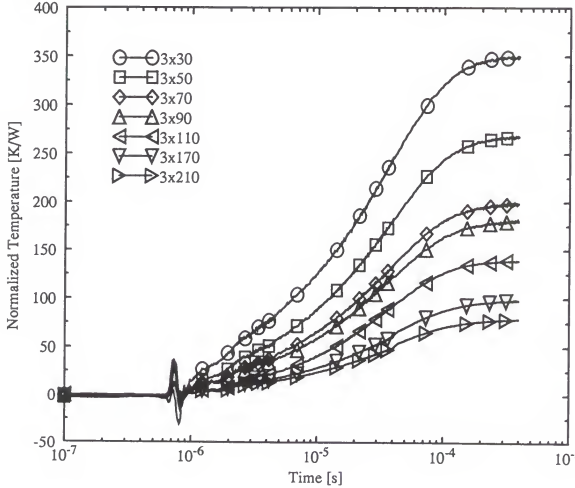

Figure 5-3 Normalized temperature response versus time of the UHF transistor for seven different emitter widths.

Thermal Impedance Model and SPICE

The above results must be compressed into a manageable model for use in circuit simulation. Using the version of SPICE modified to include a temperature (ΔT) terminal described in the previous chapter, the thermal impedance can be incorporated into precision circuit simulation of dc, ac and transient. To convert the measured step-response data to an equivalent RC-ladder model, the program TIPP was used to provide SPICE with calculated values for the thermal resistance and thermal capacitance components needed for the thermal equivalent circuit [Bro93]. TIPP can handle up to five poles for the thermal-impedance model, and input can come from either measured transient data (ΔT vs. time as found

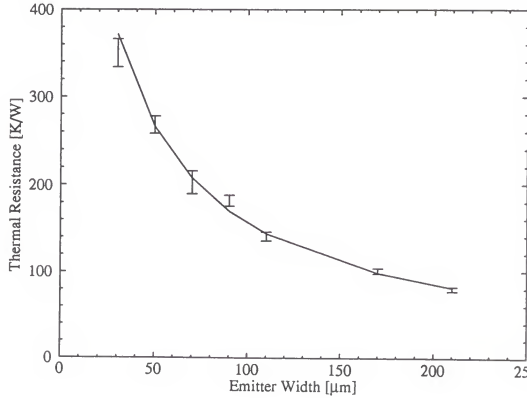

Figure 5-4 Thermal resistance versus emitter width for the UHF transistor with emitter length of 3 μm

here) or a predictive physical model [Joy70]. Figure 5-5 shows the normalized temperature response for a 3x70 μm^2 transistor and the resulting three-pole and single-pole fits. The three-pole fit is good, and generally is the best trade-off between accuracy and complexity for precision applications. For some situations the single-pole model is sufficient.

Many issues must be considered when deciding whether to use measurement or a physics-based predictive model for determining the thermal impedance in a process. Typically, manufacturers extract model parameters for each transistor in a process from measurement. The

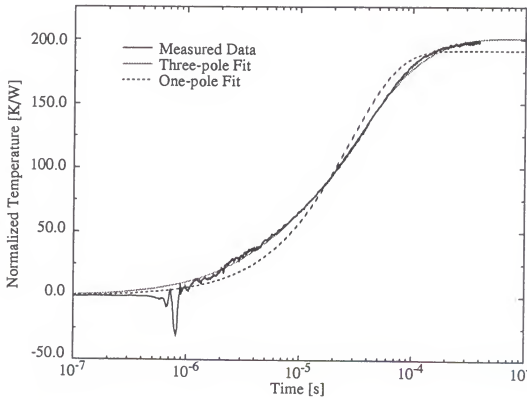

Figure 5-5 Comparison of normalized temperature response for one and two pole fits and measured data of the UHF 3x70 μm transistor

extraction is automated and can be done fairly quickly. If the measurement of the thermal impedance is included in the extraction, the time spent on each transistor is greatly increased.

Consider just the calibration step of extracting the current temperature coefficients. Estimating that each data point takes 30 s and that the minimal of two V_{CE} values per V_{BE} (which could case large errors) for two temperatures, gives a minimum of 2 minutes per V_{BE}. Add to that the time for the change of temperature of approximately 10 min and the time to find the correct operating region for the measurement. This takes about 30 min per transistor before the transient measurement has been made.

Considering the time that would be required to measure parameters for every BJT geometry, development of a model for thermal impedance dependent on process parameters that scales with transistor geometry would be preferable. Such a model should use physics as much as possible but exclusively. The trade-off here is the accuracy. A physical model cannot include all effects and still yield results quickly. For each neglected effect the accuracy usually decreases. For some structures the assumptions needed to solve the heat-flow equations may not allow enough accuracy to be useful. For these structures the loss of accuracy in the physics can usually be corrected by using empiricism and the information from measuring only a few transistors. In the case of a vertical, junction-isolated, rectangular emitter BJT good predictive physics-based models

exist (such as the one described in Chapter 2). For the wafer-bonded UHF transistors measured here, a scalable thermal model is difficult to develop due to the large process variation of the oxide depth that greatly affects the thermal impedance. For thin SOI technologies where the isolation depth and thickness are well-controlled, physical modeling seems simple until the increased importance of heat-flow out the contacts and wires is considered. Extraction of thermal impedance for some transistors in a process will always be needed either to validate a model or to tune some empirical parameters. A reasonable goal is to avoid spending needlessly long times measuring every BJT in a process.

Self-heating and Model Parameter Extraction

When standard parameter-extraction procedures are used to find parameters such as Early voltage and knee current, the data typically include some self-heating effects. When these parameters are used in a model with self-heating, the self-heating effects are overrepresented. For accurate modeling, the parameter extraction procedures themselves must be modified to separate thermal and electrical effects. This section describes a process to accurately extract these high-current parameters.

Most of the SPICE Gummel-Poon model parameters are normally extracted in low-power regions where self-heating effects can be ignored. Three parameters, however, are usually extracted in high-power regions where self-heating can be important. These are the knee current, the Early voltage, and the parasitic base resistance (R_B). I_{KF} and R_B are

extracted from a Gummel plot at high current, usually with a significant base-collector voltage to avoid quasi-saturation, so that power dissipation may not be negligible. V_A is extracted from the slope of a collector current versus collector-emitter voltage plot, which is very sensitive to the power dissipation of the measured device. It is tempting to try pulse measurements to avoid self-heating-induced problems by measuring the device before it can heat up. Unfortunately, because the thermal response in scaled devices has time constants on the order of nanoseconds, it is very difficult to measure a BJT in electrical steady-state before significant self-heating begins. In the approach presented here, the device is allowed to reach thermal steady state before data is recorded.

For comparison, a normal isothermal extraction not accounting for self-heating was done first. The model parameters were extracted and optimized in the normal way for a packaged 1.6 x 5 $(\mu m)^2$, junction isolated, NPN transistor obtained from Analog Devices [Get78]. All measurements were made using a temperature-controlled oven to supply a constant ambient temperature. The data for the characteristic curves shown in Figure 5-6 were measured from a 0-3 V sweep of the collector-emitter voltage with the base voltage held constant at 0.8, 0.85, and 0.9 V. Each point is the average of 16 measurements made after holding the bias for 20 seconds so the temperature could stabilize. The same delay-and-measure technique was used to record the Gummel plots used in the extraction. For the Gummel plots the base-emitter voltage was swept from

0.5-1.1 V with the collector-base voltage held constant at 0.5 V to avoid quasi-saturation. Note that it would not be uncommon to use a V_{CB} of 1 V, doubling the power dissipation seen here.

After the device Gummel plots and characteristic plots were measured at several temperatures, the initial parameter set was extracted using the curve-fitting techniques described by Getreu [Get78]. The exception to this was the base and emitter resistances were extracted using the method described by Ning and Tang [Nin84]. This is an adequate

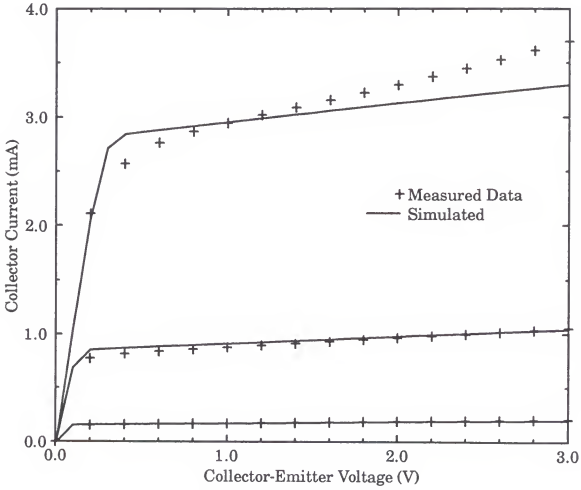

Figure 5-6 Characteristic curves resulting from a normal model parameter extraction.

method for getting values for these parasitic parameters when ac methods are unavailable. The extraction was for only the parameters needed for simulation of the forward-active and saturation regions; the reverse modes of operation were ignored because of their infrequent use in circuit design and the obvious extension from the forward modes examined here.

Next the model parameters were tuned using nonlinear optimization, the extracted data, and the output of SPICE simulations of the measurement using the parameters. Most of the parameters required only small corrections with the exception of the Early Voltage. Results of the optimization are shown in Figure 5-6. This simulation result is a poor fit to the measured data especially at high currents where the effects of self-heating can be seen in the increased slope of the I_C versus V_{CE} curve.

One way to correct for such self-heating errors is to measure the thermal resistance and use it with the self heating model in SPICE. The extracted effective thermal resistance using (5.9) was 800 K/W. To complete the self-heating model, the SPICE thermal parameters XTI and XTB were also extracted using the measured data over many temperatures. The resulting fits to the data are shown in Figure 5-7 and Figure 5-8. When self-heating is added to the simulation using the same model parameters, as shown in Figure 5-9, the conductance modeling is improved but the accuracy of the current modeling is not. It is not enough to simply include self-heating after an isothermal extraction.

The model parameters were optimized once more, this time including self-heating in the model. The Early voltage was not optimized; the value computed from (5.9) during the thermal resistance extraction was used instead. The results are shown in Figure 5-10. The fit to the data is much improved with good agreement in both slope and magnitude of the collector current.

The model parameter optimization including self-heating only affected three parameters significantly: I_{KF}, V_A, and R_B. The value of I_{KF}

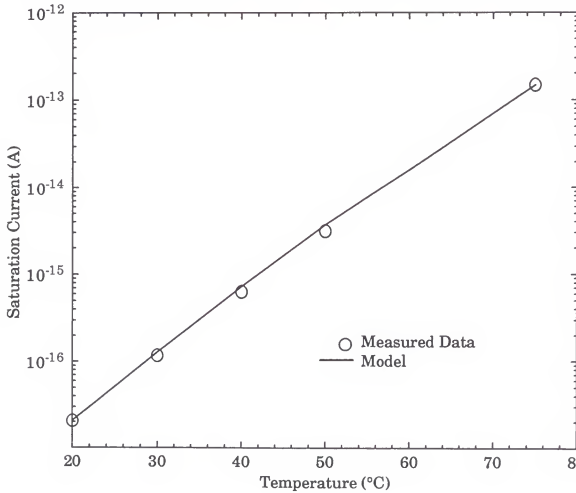

Figure 5-7 Resulting model fit for extraction of XTI from the I_S versus temperature data.

in this optimization was 40% lower than that found in the isothermal extraction. Self-heating increases the collector current in the isothermal extraction, so the electrical value of knee-current is overestimated. A 50% increase in the electrical value V_A results from taking into account the increased slope of the characteristic curves due to self-heating. Finally, a 25% decrease in the base resistance is likely caused by temperature dependence of the current gain. Although the change in base resistance

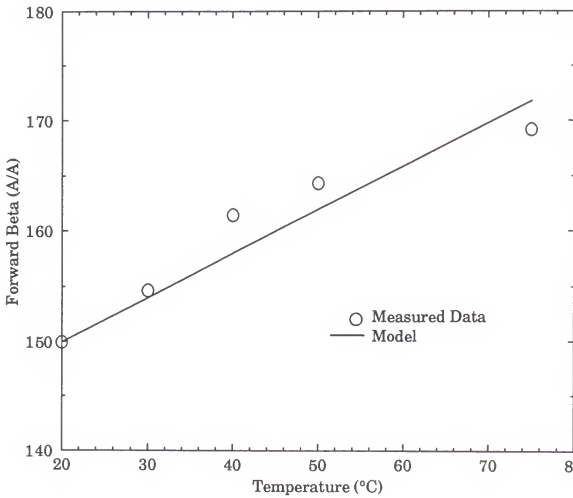

Figure 5-8 Resulting model fit for extraction of XTB from the β_F versus temperature data.

when self-heating is included, it is likely that R_B is best extracted using high-frequency ac methods [Get78].

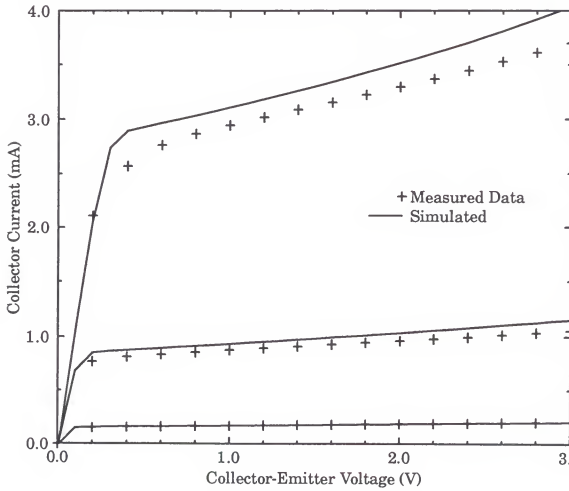

Figure 5-9 Characteristic curves resulting from the same model parameters with self-heating added to the simulation.

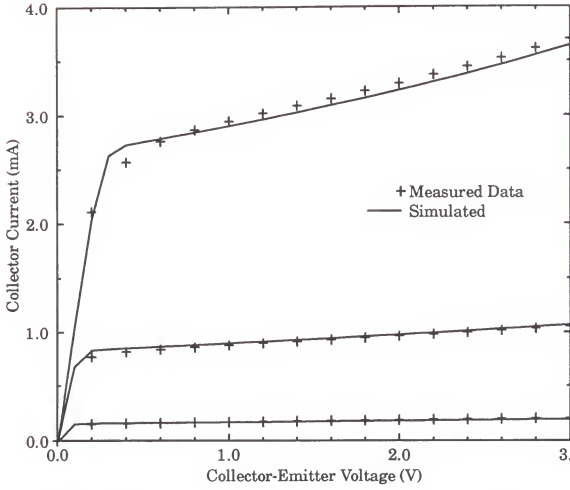

Figure 5-10 Characteristic curves resulting from including self-heating in the extraction process.

CHAPTER 6
SELF-HEATING FOR SOISPICE

SOISPICE is a simulation tool developed at the University of Florida to do physics-based modeling of silicon-on-insulator metal-oxide-silicon field-effect transistors, SOI MOSFETs, for circuit simulation [Fos96, Suh95, Yeh95]. The basic structure for these devices is shown in Figure 6-1. There are two variations of this device, differing in the thickness of the body layer. In one case the body layer is thick enough so that in normal saturation region operation the depletion region does not reach the bottom of the body, and in the other case the body is thin enough for the depletion region to reach the bottom. These structures are called either partially depleted, PD, or fully depleted, FD, SOI MOSFETs.

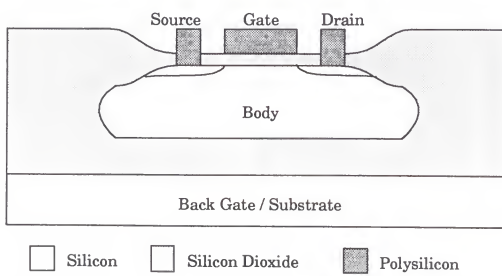

Figure 6-1 Basic structure of a SOIMOSFET with SiO2 insulating layer.

The SOI MOSFET device is surrounded on all sides with an insulating layer, usually SiO_2, that forms a barrier to heat flow. SOI devices tend to get much hotter than their bulk-silicon counterparts, because the heat is trapped in the small silicon island the device is built in [Goo95]. The dominant temperature-dependent effect on MOSFET current is the carrier mobility, which is a much weaker function of temperature than the dominant effect (carrier concentration) on BJT current. For comparable thermal impedances and power dissipations, self-heating can be ignored in a bulk MOSFET when it cannot in a BJT. Because of the insulation layer surrounding a SOI MOSFET, the thermal impedance can be one or two orders of magnitude greater than that of a comparable geometry bulk device. This increase in thermal impedance causes generation of much greater temperatures within the SOI MOSFET, and even the weak temperature dependencies of the transistor become significant.

Self-Heating Implementation

The following shows how the self-heating model was implemented into SOISPICE, and then some simple circuit simulation examples are shown. The implementation has many similarities to the BJT approach, but some major differences exist. Some of the differences result from the organization of SOISPICE. Other differences were requested by SOISPICE users, who preferred a single-pole thermal impedance, implemented internally, without additional input-file circuit elements, as

was done in the BJT implementation. The organization of the SOI MOSFET model allows all of the current and voltage derivatives to be easily computed numerically. This numerical approach was a necessity, because few of the physical model equations have closed-form solutions for their derivatives. The following discussion will focus on these features that were unique to the SOISPICE self-heating implementation. The details of the SOI MOSFET electrical modeling are still under development and are beyond the scope of this work, so they will be ignored when possible. Detailed FORTRAN code modifications to SOISPICE for this self-heating model can be found in Appendix C.

The SOI MOSFET Model

For reference, the schematic for the SOI MOSFET model is shown in Figure 6-2. The model has many current sources, but most are reactive and dissipate no real power. The dominant sources of real power are the channel current, I_{CH}, and the parasitic BJT current, I_{BJT}. The other real current sources, I_{RGt} and I_{Gi} are orders of magnitude smaller and do not contribute significantly to the power dissipation. Other sources of power dissipation include the parasitic resistances, specifically the drain and source resistances. These may not always be negligible in computing the total power dissipation, but a problem arises with the thermal impedance modeling if they are included. The thermal impedance modeling assumes the power is generated in the channel region under the gate, and not in the large adjacent source and drain regions. However, since this

implementation uses a simple, less accurate, single-pole model, including the source and drain resistances in the power computation adds more to the total accuracy than the thermal impedance modeling deficiency takes away. Therefore, these resistive power sources were included.

Specifying Self-Heating

The thermal subcircuit for the SOI MOFET is shown in Figure 6-3. The ground connection, T_A, represents the ambient temperature of the circuit defined globally in the input file. No external connections are

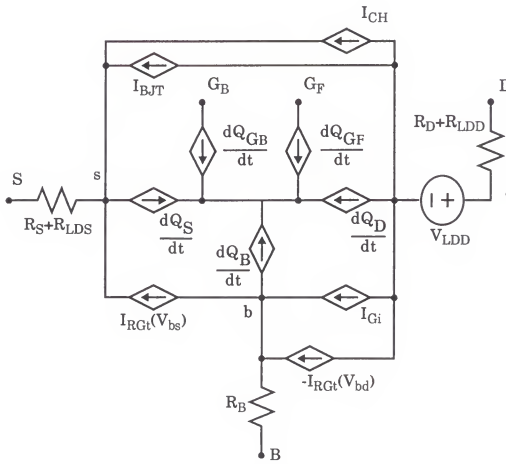

Figure 6-2 Schematic for the SOISPICE model [Fos96].

allowed to the thermal circuit in this implementation so the device description line becomes:

Zxxxx nd ngf ns ngb [nb] [BJT] model RTH= CTH= L= W= ...

where RTH and CTH are two new device-line parameters that describe the thermal subcircuit. If the parameter RTH is not specified on the device line, the thermal subcircuit is not generated, and the standard isothermal model is used. Since there are no external connections to the thermal subcircuit and the ground node is already defined, only one new node is needed to define the temperature in the self-heating model, so only one new row and column are added to the device matrix.

Computing the Derivatives

SOISPICE computes all of the current and charge derivatives with respect to node voltages numerically. The currents are computed from the model equations once at the bias point of the current iteration. For each node voltage the model is recomputed with a perturbation, and a finite-difference approximation to the derivative with respect the node voltage is computed. For self-heating this process is extended to include another

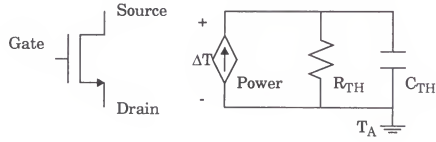

Figure 6-3 The thermal subcircuit added to the SOIMOSFET model for simulating self-heating.

perturbation computation for the temperature, and the power is added to the list of currents. For this process to have the desired results, the model equations must include a power computation and the equations for the model parameter temperature dependence.

Computing the Power Dissipation

The equation used for the power dissipation is

$$P = |I_{CH} + I_{BJT}|V_{ds} + |I_{CH} + I_{BJT}|^2(R_D + R_S + R_{LDD} + R_{LDS}), \qquad (6.1)$$

where I_{CH} is the channel current, I_{BJT} is the parasitic BJT collector current, R_D and R_S are the drain and source resistances, and R_{LDD} and R_{LDS} are the low doped drain and source region resistances. The other modeled currents that are not included in this power dissipation computation, the thermal generation and impact-ionization currents, flow in regions away from the channel and are an order of magnitude smaller than the two included currents.

Local Temperature Update of Parameters

The second task of including the temperature dependence of the model parameters is achieved using a new subroutine. This subroutine contains all of the SOI MOSFET temperature equations found in the global temperature update routines. These equations have been optimized to reduce computation time. The parameters affected are shown in Table 6-1 with a short description of each one. These dependencies were implemented for the SOISPICE global temperature model by Glen Workman [Wor96]. Equations for both the partially depleted and fully depleted devices are included in the subroutine. A flag is used to select between

Table 6-1 List of model parameters with temperature dependence.

Parameter Name	Brief Description
XNIN	Intrinsic carrier density
EGFET	Silicon energy gap
VTT	Local Thermal Voltage
XMULDSJ	LDD carrier mobility
XMUBH	Minority carrier mobility in high-doped body
UO	Zero field electron mobility
VSAT	Carrier saturated drift velocity
WKF	Front-gate work function difference
WKB	Back-gate work function difference
VFBF	Front-gate flat-band voltage
VFBB	Back-gate flat-band voltage
JRO	Body-source junction recombination current coefficient
ALPHA	Impact-ionization parameter α_0
TAUG	Carrier lifetime in lightly doped region
VBI	Source to bulk built-in potential
PHIB	Twice Fermi potential of body
RLDD	LDD sheet resistivity
BETA	β
FSAT	fsat

equations that the PD and FD devices do not have in common. The result is that only one routine must be modified if the temperature equations are modified in the future. As was the case in the BJT self-heating implementation, the thermal voltage must be treated with special care. This parameter is defined globally for all devices in SPICE, and a local

value (shown as VTT in Table 6-1) must be defined for the model equations.

Loading the Jacobian

The loading of the self-heating elements into the Jacobian matrix is done methodically. The current derivatives with respect to temperature are loaded in the new temperature column at the rows corresponding to the current branch nodes with the sign indicating direction of flow. The power derivatives with respect to voltage are loaded in the temperature row at the columns corresponding to the two nodes that comprise each voltage. The thermal conductance is loaded at the temperature diagonal element. The terms associated with the thermal capacitance are not loaded at this point, but instead are handled during preprocessing. The Jacobian matrix load for the ac analysis is implemented in nearly the same way, using complex variables for the reactive elements instead of the current-conductance pairs of the transient analysis. The simplified flowchart of how the thermal model implementation works is shown in Figure 6-4.

SOISPICE already has the capability to deal with linear capacitors. It is a burden on the programer to implement all of the state variables and computations to provide for the thermal capacitance inside the SOI MOSFET routines. By developing a way to trick SOISPICE into believing that the thermal capacitance is the same as any other capacitance, SOISPICE processes the thermal capacitance with all of the other circuit

linear capacitor elements. This is done by adding a feature to the SOISPICE FIND subroutine. The last argument of the subroutine call, IFORCE, is used either to return the location of a device in memory or create space in memory and store a device there. The latter option checks whether the device has an unique name and returns an error if it does not. A third option was added that stores the device without checking the name uniqueness. Using this option the thermal capacitors are all added to the

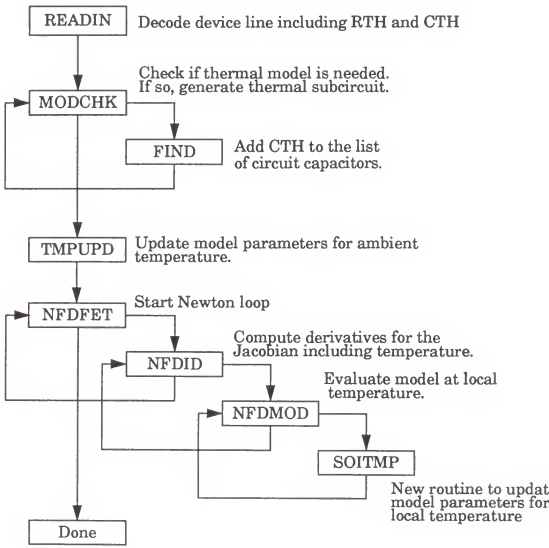

Figure 6-4 Flowchart of SIOSPICE execution showing processing done for self-heating.

bottom of the normal capacitor list with the proper connections to their thermal subcircuits, all having the same name to avoid possible name collisions with the regular circuit capacitors. SOISPICE then processes the thermal capacitors like any other linear capacitor in the circuit without adding more code to the SOI MOSFET model routines.

This approach also improves efficiency. Since any computations added to the model routines are executed twice per iteration to compute the partial derivatives, excluding the computations associated with thermal capacitance from the model routines reduces computational overhead.

Convergence Issues

Because of floating-body effects and other complex SOI MOSFET effects, convergence control and initial conditions must be set with care even without self-heating to achieve convergence of some circuits. Self-heating tends to degrade the robustness of convergence, but this is expected any time the dimension of the Jacobian is increased as it is for self-heating. The difficulty of estimating the extent of the degradation is exacerbated by the convergence behavior of the isothermal model, which was being refined concurrently with the self-heating implementation. Either the technique of skipping the first Newton iteration as used in the BJT implementation or a traditional temperature step-size limiting algorithm could be incorporated to enhance convergence.

Simulation Results

Three example circuits were simulated to illustrate some of the effects of self-heating. The first is a dc simulation to generate the characteristic curves. The common-source configuration is used with the source-drain voltage swept from 0 to 3 V for three different gate voltages. The results for the floating body case are shown in Figure 6-5.

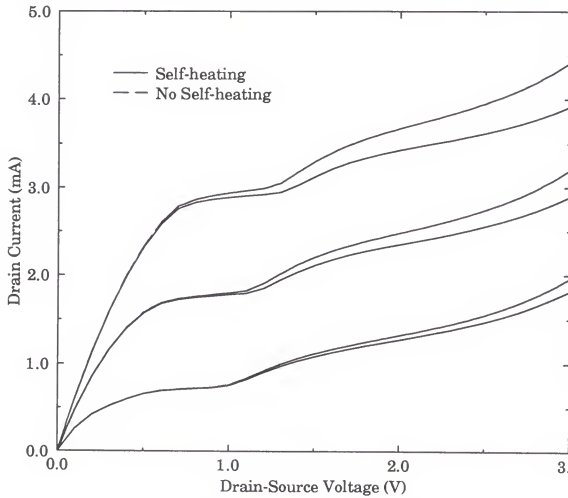

Figure 6-5 DC-sweep simulation for an n-channel, floating-body, PD, SOI MOSFET with 0.2 μm length and 10 μm width. V_{DS} was swept from 0 to 3 V for V_G of 1, 1.5 and 2 V. The thermal resistance was 5 kW/K

The second is a step-response test of a single n-channel, PD, SOI MOSFET. Again, the common-source configuration is used with the gate biased for the saturation region. The drain voltage is then pulsed. The resulting drain current response is shown in Figure 6-6. Unlike the BJT behavior, the current decreases from the effects of self-heating. The dominant temperature-dependent carrier mobility decreases with temperature and the drain current along with it. Other than the obvious

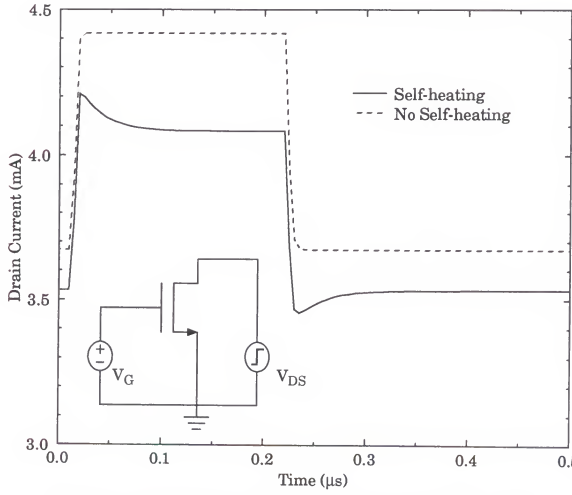

Figure 6-6 Transient current response to a drain voltage step for a common-source, n-channel, 0.2x10um, PD, SOI MOSFET. Vg=2V Vds=2-3V Rth=5kW/K

biasing errors that can result, this negative temperature coefficient can cause hysteresis in the dc transfer characteristic when the thermal resistance-drain current product reaches a critical value [Fox93a].

The third example demonstrates that self-heating is not a significant problem for complementary MOS logic circuits; the primary application for SOI MOSFETs currently. The circuit for a clocked complementary SOI MOSFET inverter is shown in Figure 6-7. Figure 6-8 shows the output voltage waveform and the temperature waveform of the n-channel device. Notice that the temperature increases in steps synchronized with the switching of the inverter. The only time a complementary gate dissipates significant power is during switching. This is shown as a steep step in the temperature waveform, and a slow cooling until the next transition. The simulation shows the temperature

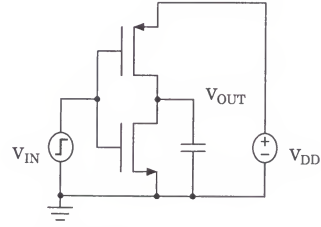

Figure 6-7 Complementary inverter circuit used for dynamic power dissipation example. V_{IN} is pulsed from 0-3 V at a frequency of 40 MHz and V_{DD} is 3 V.

approaching a steady-state value. The final value of the temperature is a function of the total dynamic power dissipation for the inverter:

$$P_D = \frac{C_L(V_{DD})^2}{\tau} \tag{6.2}$$

where C_L is the load capacitance, V_{DD} is the gate bias voltage, and τ is the period of the clocking signal. For the inverter used here,

$$P_D = \frac{0.5\text{pF}(3\text{V})^2}{25\text{ns}} = 18\mu\text{W} . \tag{6.3}$$

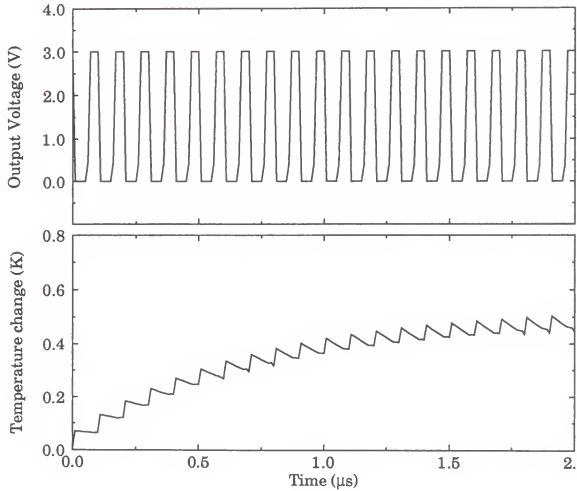

Figure 6-8 Inverter output (top) and temperature change (bottom).

Since P_D is the total power for the inverter, half of the power is dissipated in each transistor. Multiplying half the result of (6.3) by the thermal resistance of 5 kK/W gives about 0.5 K increase in temperature, which is consistent with the simulation steady-state temperature. Such a temperature increase is too small to significantly affect the circuit performance, which is highly typical of self-heating in CMOS digital circuits. This is not to say that thermal effects cannot be important in digital circuits. The heat generated must ultimately leave the package, and for large, dense circuits the thermal impedance of the die-to-package-to-ambient is important for reliability, since it controls the rise in average temperature. However, self-heating is usually insignificant, since the instantaneous device temperature does not follow the instantaneous power dissipation.

Conclusion

A self-heating implementation was shown for the SOI MOSFET models in SOISPICE. A single-pole thermal subcircuit was added to model the power dissipation through the thermal impedance of the device. Three example simulations were shown. The step response showed how the negative temperature coefficient of the current affects biasing and other analog applications. The inverter simulation demonstrated CMOS logic circuitry where the contribution of self-heating to the device temperature is insignificant. The greatest impact of self-heating in SOI MOSFET circuits will likely be on analog circuits and in parameter extraction.

CHAPTER 7
CONCLUSION

An extensive study of self-heating modeling and effects has been presented here. A historical look at self-heating for circuit simulation showed some continuing deficiencies in modeling. The quality of the modeling has been improved by this work in several areas. Three implementations of self-heating in the popular circuit simulator SPICE were shown: two for BJTs and one for PD and FD, SOI MOSFETs. Circuit effects of self-heating were examined for dc, ac, and transient analysis. A new measurement technique for extraction of thermal parameters was presented.

The two approaches to implementing self-heating in the Gummel-Poon, BJT model each had their advantages. The implicit temperature approach did not require any changes to the Jacobian matrix. This approach works well for dc and ac analyses, allowing physically accurate modeling of the thermal impedance. The inability of this implementation to simulate transient analyses necessitated finding a different approach that could. The second approach enabled transient analysis by treating the power dissipation as an electrical analog. The power and thermal impedance formed a thermal subcircuit that allowed explicit computation of a temperature increase. The thermal subcircuit required extending the

device matrix to include the explicit temperature node. Because the thermal impedance is modeled with an RC network, a trade-off between accuracy and complexity was required. The circuit application determines the level of accuracy needed.

The last implementation for the SOISPICE models PD and FD, SOI MOSFETs used a single-pole thermal subcircuit model to model self-heating. The simplicity of the thermal model is reflected in its easy use, requiring only two new parameters on the device line. A numerical approach to computing the derivatives for all of the elements of the Jacobian made the addition the matrix node for temperature easier than the previous implementations for BJTs.

An examination of the effects of self-heating on circuit performance was presented that included techniques for analyzing these effects in hand calculations. Simulations of a common-emitter amplifier showed the ac and dc effects on open-loop gain and the transfer function distortion for BJTs. The pin-driver circuit demonstrated how long thermal settling times can occur in switching applications. The SOI MOSFET simulations showed that care must be used in analog circuits and that digital applications are not directly affected by self-heating.

A new measurement technique for extracting the thermal impedance of BJTs was shown. The two-step procedure used a dc calibration step to extract the current temperature coefficients, and a high- speed measurement of the base current step response to extract the

temperature waveform. The results of the first step are then used to scale the base current into temperature. After normalizing for power, the temperature response is fitted with a multiple-pole model for use in the thermal subcircuit.

The effects of self-heating on Gummel-Poon model parameter extraction was examined by including the self-heating model in the optimization simulation. The Early voltage, knee current, and base resistance were shown to be the parameters affected most by self-heating.

Self-heating is a potentially important effect that all circuit designers should be conscious of in their designs. It can have devastating effects on some precision circuits if not accounted for before manufacturing. Self-heating is not an important factor in every circuit, so designers should use discretion when using it in simulation. There are penalties in simulation time and convergence if self-heating is applied carelessly.

APPENDIX A
SPICE 2G.6 CODE CHANGES FOR MODELING OF SELF-HEATING USING THE IMPICIT TEMPERATURE APPROACH

The following is a summary of the changes made to SPICE 2G.6 to implement self-heating in the BJT model. The local temperature of each device is computed for each iteration as a function of the power dissipation in the collector-base space-charge region (SCR). Temperature is not treated as a state variable, and does not require addition of a new row and column in the circuit matrix for each BJT.

Data structure expansion for new model parameters

The SPICE data structure must be expanded to include the new model parameters, state variables, and device variables necessary for the thermal model. A subset of the following six new model parameters are needed to specify the thermal model shown in Table A-1.

Table A-1 Added model paramters.

RTH	Thermal resistance
CTH	Thermal capacitance
NEPI	Epi-layer doping concentration
LTH	Emitter length
WTH	Emitter width
DTH	Collector-base junction depth

RTH and CTH combine for the single-pole model for the thermal impedance, ZTH, or ZTH can be computed from RTH using a frequency-domain model for a point heat source. The last four parameters can be used to predict RTH from the device bias, doping, and geometry, or RTH can be supplied directely. To simplifiy future modifications, eight new model parameter positions were created, with the two extra parameters left undefined.

Two variables were added to the state vector shown in Table A-2.

Table A-2 State-varaiables.

TEMPO	Device local temperature change
RTHO	Device thermal resistance

Since these are computed "on the fly" it is helpful to know what their values were in the previous iteration, especially in dc sweeps.

Six variables were added to the device shown in Table A-3.

Table A-3 Device variables.

GPIE	electrical-only base-emitter conductance
GMUE	electrical-only base-collector conductance.
GME	electrical-only base-collector transconductance.
GOE	electrical-only collector-emitter conductance.
TCB	temperature coefficient of base current.
TCC	temperature coefficient of collector current.

These operating-point values are used in the ac analysis.

Changes to the Preprocessing and Postprocessing Routines

First the data structure must be expanded to include the parameters described above: eight model-card variables, two state variables, and six device variables.

Subroutine ADDELT

This routine adds an element and defines the size of linked-list arrays for computation of offsets. Add six device variables and eight model variables.

```
DATA
LVAL(12) = 4 → LVAL(12) = 10
LVAL(22) = 55 → LVAL(22) = 63
```

Subroutine ALTER.f

This routine changes the elements or device parameters for multiple SPICE runs. Add six device variables and eight model variables.

```
DATA
LVAL(12) = 4 → LVAL(12) = 10
LVAL(22) = 55 → LVAL(22) = 63
```

Subroutine DCOP.f

This routine prints out operating-point information. The following changes allow thermal resistance and local temperature to be printed in response to the .OP command.

```
      DIMENSION
2     CCS(12), CBX(12), RX(12),TEMP(12), RTH(12)

      DATA ATEMP, ARTH / 4HTEMP, 3HRTH /
```

Change BJT section to:

```
...
      BETADC(KNTR)=CC(KNTR)/DSIGN(DMAX1(DABS(CB(KNTR)),1.0D-20),
```

```
     1          CB(KNTR))
                TEMP(KNTR)=VALUE(LOCT+19)
                RTH(KNTR)=VALUE(LOCT+20)
                IF (MODEDC.NE.1) GO TO 325
     ...
                IF (VALUE(LOCT+20) .EQ. 0.0D0) GO TO 355
                WRITE (IOFILE,AFMT2) ATEMP,(TEMP(I),I=1,KNTR)
                WRITE (IOFILE,AFMT2) ARTH,(RTH(I),I=1,KNTR)
355             IF (MODEDC.NE.1) GO TO 360
```

These thermal variables are printed only if a thermal model is used.

Subroutine FIND.f

This routine presets storage for each input element. Add six device variables and eight model variables.

```
      DATA
LVAL(12) = 4 → LVAL(12) = 10
LVAL(22) = 55 → LVAL(22) = 63
```

Subroutine MATPTR.f

This routine builds the sparse matrix. The offset for the LXI state-vector is increased by two (to make room for RTHO and TEMPO). In the section for the BJT matrix setup:

```
NXTRM=NXTRM+19 → NXTRM=NXTRM+21
```

Subroutine MODCHK.f

This routine preprocesses device model parameters and prints out a device summary. Add the eight new model variables, their names, and their default values.

```
DIMENSION
AMPAR(115) → AMPAR(123)
DEFVAL(115) → DEFVAL(123)
IFMT(115) → IFMT(123)
IVCHK(115) → IVCHK(123)
```

IPAR holds the model parameter offset for each device type. Since the BJT is the second device in the list, the third and greater offsets must be increased by eight.

```
      DATA
      IPAR / 0, 14, 60, 72, 115 / → IPAR / 0, 14, 68, 80, 123, /
      AMPAR / ...
7     6HXTI, 6HKF, 6HAF, 6HFC, 6HRTH, 6HCTH, 6HNEPI,
8     6HLTH, 6HWTH, 6HDTH, 6H0, 6H0, ... /
```

DEFVAL holds the default values, and the new model parameters are set to zero.

```
      DEFVAL / ...
3     ...
4     1.11D0, 3.0D0, 0.0D0, 1.0D0, 0.5D0, 8*0.0D0,
1     ... /
```

IFMT holds the output format controls.

```
      IFMT / ...
2     ...
2     1, 1, 2, 0, 0, 0, 0, 2, 1, 1, 2, 1, 1, 2, 2, 2, 4, 4, 4, 4, 4, 4, 0, 0,
3     ... /
```

IVCHK defines whether the parameter can be negative.

```
      IVCHK / ...
2     ...
2     0,0,0,0,0,0,0,0,0,0,0,0,-1,0,0,0, 0, 0, 0, 0, 0, 0, 0, 0,
3     ... /
```

SPICE requires several computed model parameters to be stored at the end of the model parameter list; the thermal parameters are inserted before these. Change all BJT model-variable references (LOCV + x) where x is greater than 46 to (LOCV + x + 8), e. g.

```
VALUE(LOCV+50)=FC*PC → VALUE(LOCV+58)=FC*PC
```

Subroutine READIN.f

This routine processes element and device model definitions. Add the eight new model variables.

```
      DIMENSION
      AMPAR(115) → AMPAR(123)

      DATA
      IPAR / 0, 14, 60, 72, 115, / → IPAR / 0, 14, 68, 80, 123 /
      AMPAR / ...
     7  6HXTI, 6HKF, 6HAF, 6HFC, 6HRTH, 6HCTH, 6HLTH
     8  6HWTH, 6HDTH, 6H0, 6H0, ... /
```

Subroutine BJT.f

This routine performs the initialization, matrix load, voltage limiting, and determines the operating point currents and derivatives for the dc and transient analyses.

Add thermal resistance and temperature to the state-vector.

```
      DIMENSION ...
     2  ... CEXBC(1), GEQCBO(1), TEMPO(1), RTHO(1)
      EQUIVALENCE ...
     6  ... (GEQCBO(1),VALUE(19)),
     7  (TEMPO(1), VALUE(20)), (RTHO(1), VALUE(21)),
```

DC model parameters section

Additional needed variables.

```
      XJRB=VALUE(LOCM+17)*AREA
C...THERMAL MODEL PARAMETERS
      PC=VALUE(LOCM+30)
      XTB=VALUE(LOCM+41)
      XTB=VALUE(LOCM+41)
      EG=VALUE(LOCM+42)
      XTI=VALUE(LOCM+43)
      RTH=VALUE(LOCM+47)
      ONEPI=VALUE(LOCM+49)
      OLTH=VALUE(LOCM+50)
      WTH=VALUE(LOCM+51)
      DTH=VALUE(LOCM+52)
```

VTT=VT (so Vt can vary with each device's temperature)

Change VALUE(LOCM + x) where x > 46 to VALUE(LOCM + x + 8), e. g.

VCRIT=VALUE(LOCM+54) → VCRIT=VALUE(LOCM+62)

Thermal model flag setup

A multi-valued flag, ITHERM, to direct control for different thermal models is created with the values shown in Table A-4.

Table A-4 Key to thermal model flag: ITHERM

1	Thermal model using RTH.
2	Thermal model using predictive model (Nepi, L, W, D).
3	No thermal model.
4	Invalid region and thermal model using RTH.
5	Invalid region and predictive thermal mode.
6	Invalid region and no thermal model.

Note that ITHERM is augmented by three when the current is non-positive (CCHAT <= 0 or CC <= 0). The thermal model is not valid and should be avoided in these cases. After DC MODEL PARAMETERS and before INITIALIZATION insert:

```
ITHERM=3
IF (RTH.NE.0.0D0) ITHERM=1
IF (ONEPI.NE.0.0D0) ITHERM=2
```

Device temperature, thermal resistance, and initial current

The nominal temperature is referenced as VALUE(ITEMPS+ITEMNO)+CTOK in Kelvin. It is always the current

ambient temperature whether it is the default temperature, .TEMP value, or TNOM value. In the DC MODEL PARAMETERS section:

```
TNOM = VALUE(ITEMPS+ITEMNO)+CTOK
```

At INTIALIZATION (INITF = 2) after line 25 insert:

```
IF(ITHERM.EQ.3) GO TO 300
TEMPO(LXO+LOCT) = TNOM
IF(ITHERM.EQ.1) RTHO(LX0+LOCT)=RTH
IF(ITHERM.GT.1) RTHO(LX0+LOCT)=0.0D0
```

This initializes the temperature to TNOM on the first iteration.

THAT is the temperature predicted for the current iteration. Ideally this is computed from CCHAT, CBHAT, RTH, VBE, VBC, and TNOM, but CCHAT and CBHAT are not always valid. In the cases when the predicted currents are not available, the temperature at the end of the previous iteration is used for THAT.

Near line 50 (INITF=4):

```
IF(ITHERM.EQ.3) GO TO 210
THAT=TEMPO(LX0+LOCT)
REGION=CCO(LX0+LOCT)
IF(REGION.LE.0.0D0) ITHERM=ITHERM+3
GO TO 210
```

At end of COMPUTE NEW NONLINEAR BRANCH VOLTAGES section:

```
REGION=CCHAT
CBHAT= ...
IF(REGION.LE.0.0D0) ITHERM=ITHERM+3
IF(ITHERM.LT.3) THAT=RTHO(LX0+LOCT)
1       *(DABS((VBE-VBC)*CCHAT)+DABS(VBE*CBHAT))+TNOM
```

Flow control

To prevent the first convergence check when a thermal model is in use, the first line in BYPASS IF SOLUTION HAS NOT CHANGED section is changed to:

```
IF (INITF.EQ.6.OR.ITHERM.NE.3.OR.ITHERM.NE.6) GO TO 200
```

This avoids premature convergence in dc sweeps.

Update of temperature-dependent variables

Temperature dependent model parameters IS, ISE, ISC, BF, BR, and VT are recomputed to reflect the local temperature.

The following statements were added after the LIMIT NONLINEAR BRANCH VOLTAGES section:

```
C.. THERMAL MODEL TEMPERATURE UPDATE
210     IF(ITHERM.GT.2) GO TO 300
        RATIO=THAT/TNOM
        RATLOG=DLOG(RATIO)
        DTEMP=RATIO-1.0D0
        VTT=BOLTZ*THAT/CHARGE
        FACTLN=DTEMP*EG/VTT+XTI*RATLOG
        FACTOR=DEXP(FACTLN)
        CSAT=CSAT*FACTOR
        BFACTR=DEXP(XTB*RATLOG)
        BFM=BFM*BFACTR
        BRM=BRM*BFACTR
        C2=C2*DEXP(FACTLN/VALUE(LOCM+7))/BFACTR
        C4=C4*DEXP(FACTLN/VALUE(LOCM+13))/BFACTR
```

These equations are the same as those in the temperature update routine, TMPUPD.f.

Thermal modifications to the conductances

In the section following the CHECK CONVERGENCE section and before storage of the state-vector variables line 750 is modified and the following code is inserted to correct the intrinsic device conductances for thermal effects:

```
C
C     MODIFY CONDUCTANCES FOR THERMAL EFFECTS
C
750       IF(ITHERM.GT.3.AND.CC.GT.0.0D0) ITHERM=ITHERM-3
          IF(ITHERM.GT.2) GO TO 780
          POWER = DABS(CC*(VBE-VBC))+DABS(CB*VBE)
          TNEW = RTHO(LX0+LOCT)*POWER+TNOM
          GO TO (755,752),ITHERM
```

If necessary RTH is predicted from VCB and the device's geometry. Here the temperature dependance of the thermal conductance, OKTH, has been added,

```
C..PREDICT RTH
752       ARG1=DSQRT(WTH*OLTH)
          RATIO=WTH/OLTH
          ARG2=PC-VBC
          ARG2=DSQRT(2.0D-6*EPSSIL*ARG2/(CHARGE*ONEPI))/ARG1
          ARG3=DTH/ARG1
          F1=(0.058D0*ARG3+0.14D0)*ARG2+0.34D0*ARG3+0.28D0
          F2=0.98D0+0.043D0*RATIO-6.9D-4*RATIO**2.0D0
     1          +3.9D-6*RATIO**3.0D0
          OKTH=2.99D4/(TNEW-99.0D0)
          RTHO(LX0+LOCT)=1.0D0/(TWOPI*OKTH*2.0D0*ARG1*F1*F2)
```

The equations for modifying the conductances apply to the common-emitter g-parameters, so the hybrid-π parameters must be converted to g-parameters, modified, and converted back to hybrid-π parameters.

```
C..CONVERT HYBRID-PI TO G-PARAMETERS
755       G11 = GPI+GMU
          G12 = -GMU
          G21 = GM-GMU
          G22 = GO+GMU
```

TNEW is the temperature computed from the actual currents, not the linearly predicted ones, TCB is the temperature coefficient of base current with VBE held constant, and TCC is the temperature coefficient of collector current with VBE held constant.

```
C..COMPUTE TEMPERATURE COEF.
        TIBE1 = (XTI+(EG-VBE/VALUE(LOCM+3))/VTT)
        ARG1 = BFM*CBEN/VALUE(LOCM+7)
        ARG2 = ARG1*(XTI-VALUE(LOCM+7)*XTB-VBE/VTT)
        TCB = ((TIBE1-XTB)*CBE-ARG2)/(TNEW*BFM*CB)
        TCC = TIBE1*(1.0D0-CC*OIK*Q1/DSQRT(1.0D0+4.0D0*CBE*OIK))/TNEW
```

Since the same equations hold in the ac analysis for the complex Y-parameters, the unmodified values are saved. These electrical-only values will be corrected in the ac matrix load routine, ACLOAD.f.

```
C..SAVE FOR AC ANALYSIS
        VALUE(LOCV+4)=GPI
        VALUE(LOCV+5)=GMU
        VALUE(LOCV+6)=GM
        VALUE(LOCV+7)=GO
        VALUE(LOCV+8)=TCB
        VALUE(LOCV+9)=TCC
```

Next the g-parameters are modified and converted back to hybrid-π form.

```
C..ADJUST FOR THERMAL EFFECTS
        ARGC = TCC * RTHO(LX0+LOCT)
        ARGB = TCB * RTHO(LX0+LOCT)
        DENC = 1.0D0 - ARGC * POWER
        DENB = 1.0D0 - ARGB * POWER
        GNUM11 = ARGB * CB * CB
        GNUM21 = CB * CC
        GNUM12 = ARGB *GNUM21
        GNUM21 = ARGC *GNUM21
        GNUM22 = ARGC * CC * CC
        G11=(G11+GNUM11)/DENB
        G12=(G12+GNUM12)/DENB
        G21=(G21+GNUM21)/DENC
        G22=(G22+GNUM22)/DENC
C..CONVERT BACK TO HYBRID-PI PARAMETERS */
        GPI = G11 + G12
        GMU = -G12
```

```
GM = G21 - G12
GO = G22 + G12
```

Changes for AC analysis

Changes to the ac analysis simulation code are similar to the corrections to the conductances in the dc analysis. The only difference is the conductances become complex admittances and the thermal resistance becomes a complex thermal impedance, ZTH. The thermal impedance is computed from the single-pole model if CTH is specified on the model card. Otherwise ZTH is computed from RTH using the point heat-source model.

Routine ACLOAD.f

The following code adds two new temperature dependancies: OKTH is the thermal conductivity of silicon and OKAPPA is the thermal diffusivity of silicon. Simple relations were used to approximate the measured data for these variables [May67].

In the BJT section, check for a non-zero RTH in the state-vector and for positive collector current.

```
        GEPR=VALUE(LOCM+19)*AREA
C..THERMAL MODEL
        CC=VALUE(LOCT+2)
        RTH=VALUE(LOCT+20)
        IF(RTH.EQ.0.0D0.OR.CC.LE.0.0D0) GO TO 262
        CB=VALUE(LOCT+3)
        POWER=DABS(CC*(VALUE(LOCT)-VALUE(LOCT+1)))
             +DABS(CB*VALUE(LOCT))
        TEMP=VALUE(LOCT+19)
        GPI=VALUE(LOCV+4)
        GMU=VALUE(LOCV+5)
        GM=VALUE(LOCV+6)
        GO=VALUE(LOCV+7)
        TCB=VALUE(LOCV+8)
        TCC=VALUE(LOCV+9)
        GO TO 263
```

If no thermal model,

```
262     GPI=...
```

Excess phase and imaginary parts of admittances,

```
263     XGM=...
270     GX=...
```

Modify admittances

```
C..THERMAL MODIFICATIONS TO ADMITTANCES
        IF(RTH.EQ.0.0D0.OR.CC.LE.0.0D0) GO TO 280
C..CONVERT TO COMMON-EMITTER Y-PARAMETERS
        Y11=CMPLX(GPI+GMU,XCMCB+XCPI+XCMU)
        Y12=CMPLX(-GMU,XCMCB+XCMU)
        Y21=CMPLX(GM-GMU,XGM-XCMU)
        Y22=CMPLX(GO+GMU,XCMU)
```

Single-pole model,

```
        IF (VALUE(LOCM+48).EQ.0.0D0) GO TO 273
        ZTH=CMPLX(RTH,VALUE(LOCM+48))
        GO TO 275
```

Computed from RTH using the temperature dependance of the thermal conductance and diffusivity, OKAPPA [Nis80].

```
C..COMPUTE ZTH FROM RTH
273     OKTH=2.99D2/(TEMP-99.0D0)
        OKAPPA=OKTH*(TEMP-99.0D0)/(2.33D0*(TEMP-159.0D0))
        ARG1=DSQRT(OMEGA/(2*OKAPPA))
        REFF=1.0D0/(2.0D0*TWOPI*OKTH*RTH)
        PHS=-REFF*ARG1
        OMAG=RTH*DEXP(PHS)
        ZTH=CMPLX(OMAG*DCOS(PHS),OMAG*DSIN(PHS))
```

Correct electrical-only admittances for self-heating,

```
C..MODIFY
275     DENB=1.0D0/(1.0D0-TCB*POWER*ZTH)
        DENC=1.0D0/(1.0D0-TCC*POWER*ZTH)
        ARG1=CC*CB
        Y11=(Y11+TCB*ZTH*CB*CB)*DENB
        Y12=(Y12+TCB*ZTH*ARG1)*DENB
        Y21=(Y21+TCC*ZTH*ARG1)*DENC
        Y22=(Y22+TCC*ZTH*CC*CC)*DENC
C..CONVERT BACK TO HYBRID-PI
        GPI=REAL(Y11)+REAL(Y12)
        GMU=-REAL(Y12)
        GM=REAL(Y21)-REAL(Y12)
```

```
GO=REAL(Y22)+REAL(Y12)
XGM=AIMAG(Y21)+AIMAG(Y22)
XCPI=AIMAG(Y11)-AIMAG(Y12)
XCMU=AIMAG(Y22)
XCMBC=AIMAG(Y12)-AIMAG(Y22)
```

Stuff into matrix as usual,

```
280     LOCY=LYNL+NODPLC(LOC+24) ...
```

APPENDIX B
SPICE 2G.6 CODE CHANGES FOR MODELING OF SELF-HEATING USING AN EXPLICIT TEMPERATURE NODE

The following is a summary of the changes made to SPICE 2.G6 to implement self-heating in the BJT model. The local temperature of each device is computed for each iteration as a function of the power dissipation in the collector-base space-charge region (SCR). In the approach used here, temperature is represented explicitly by the addition of a new node to the device model, and hence a new row and column are needed in the device matrix equations. The dc, ac and transient analyses are all implemented using this thermal node approach.

Data structure expansion

The SPICE data structure must be expanded to include the new state variables and matrix locations for the thermal model. Five variables were added to the state vector shown in

Table B-1 State-variables.

VTMPO	Device local temperature change.
GTCO	Collector thermal conductance.
GTBO	Base thermal conductance.
CPO	Device instantaneous power.

Note: if more than a single-pole model is desired to model the thermal response then an additional energy and energy-current state-variable will be needed for each additional pole.

Subroutine Modifications

What follows is a quick description of the changes needed to each routine to implement the thermal model. First the data structure must be expanded to include the parameters described above. This includes eight matrix locations and four state variables.

ADDELT

This routine adds an element and defines the size of linked-list arrays for computation of offsets. Add eightelements to the matrix and one variable to the device.

```
DATA
LNOD(12) = 38 ⟶ LNOD(12) = 46
LVAL(12) = 4 ⟶ LVAL(12) = 5
NNODS(12) = 4 ⟶ NNODS(12) = 5
```

ALTER

This routine changes the elements or device parameters for multiple SPICE runs. Add eight elements to the matrix and one variable to the device.

```
DATA
LNOD(12) = 38 ⟶ LNOD(12) = 46
LVAL(12) = 4 ⟶ LVAL(12) = 5
```

ELPRNT

In statement 300 change:

```
NODPLC(LOC+36) ⟶ NODPLC(LOC+44)
```

ERRCHK

Add an external node to the BJT device line:

```
DATA NNODS(12) = 4 → NNODS(12) = 5
```

FIND

This routine presets storage for each input element. Add eight elements to the matrix and one variable to the device.

```
DATA
LNOD(12) = 38 → LNOD(12) = 46
LVAL(12) = 4 → LVAL(12) = 5
```

MATLOC

In statement 830:

```
NODPLC(LOC+36) → NODPLC(LOC+44)
```

Then add new elements to matrix:

```
  NODET=NODPLC(LOC+36)
...
  NODPLC(LOC+37)=INDXX(NODE4,NODET)
  NODPLC(LOC+38)=INDXX(NODE5,NODET)
  NODPLC(LOC+39)=INDXX(NODE6,NODET)
  NODPLC(LOC+40)=INDXX(NODET,NODE4)
  NODPLC(LOC+41)=INDXX(NODET,NODE5)
  NODPLC(LOC+42)=INDXX(NODET,NODE6)
  NODPLC(LOC+43)=INDXX(NODET,NODET)
```

NODET is the external thermal node.

MATPTR

This routine builds the sparse matrix and it is here that the new nodes are added.

In statement 530 change:

```
NODPLC(LOC+36) → NODPLC(LOC+44)
```

Then add the new nodes and elements to the matrix:

```
NODET=NODPLC(LOC+36)
CALL RESERV(NODE4,NODET)
CALL RESERV(NODE5,NODET)
CALL RESERV(NODE6,NODET)
CALL RESERV(NODET,NODE4)
CALL RESERV(NODET,NODE5)
CALL RESERV(NODET,NODE6)
CALL RESERV(NODET,NODET)
```

The offset for the LXI state-vector is increased by 4 (to make room for state variables). In the section for the BJT matrix setup:

NXTRM=NXTRM+19 → NXTRM=NXTRM+23

MODCHK

This routine preprocesses device model parameters and prints out a device summary. In statement 710 change:

NODPLC(LOC+36) → NODPLC(LOC+44)

then move the reference to the new external node:

NODPLC(LOC+36)=NODPLC(LOC+6)

READIN

This routine processes element and device model definitions.

```
DATA
L2NOD(12) =36 → L2NOD(12) =44
```

To read the external thermal node, above statement 306 change:

GO TO 308 → GO TO 307

then after 306 add:

```
307     IF(NODPLC(ICODE+6).NE.0) GO TO 308
        IFLD=7
        IF (NODPLC(LOC+L2NOD(ID)).NE.0) GO TO 309
        NODPLC(LOC+6)=VALUE(IFIELD+6)
        GO TO 308
309     NODPLC(LOC+6)=0
308     CONTINUE (no change)
```

The thermal node is temporarily placed in LOC+6 for reference in TOPCHK and is moved to LOC+36 in the MODCHK routine

TOPCHK

Add external thermal node reference:

```
DATA NNODS(12) = 5
```

TRUNC

In statement 80 change:

NODPLC(LOC+36) → NODPLC(LOC+44)

Changes for DC analysis

The following are changes needed for operating-point and dc-sweep analysis. Here we have created a mirror BJT routine that is executed in the event of a thermal analysis. This eliminates all of the "if thermal then do this" statements that would otherwise be needed in the original BJT routine. A drawback is the extra maintenance of a new model routine.

BJT

This routine performs the initialization, matrix load, voltage limiting, and determines the operating point currents and derivatives for the dc and transient analyses.

In the case of a thermal transistor branch to thermal version after LOCM assignment:

```
IF (VALUE(LOC+36).NE.1) THEN
    CALL BJTT(LOC)
    LOC=NODPLC(LOC)
    GO TO 10
ENDIF
```

BJTT

This routine performs the initialization, matrix load, voltage limiting, and determines the operating-point currents and derivatives for the dc and transient analyses for thermal transistors.

Add thermal resistance, temperature, thermal conductance, and power current to the state-vector.

```
      DIMENSION ...
3     VTMPO(1), GTCO(1), GTBO(1), CPO(1)
      EQUIVALENCE ...
7     (VTMPO(1),VALUE(20)), (GTCO(1),VALUE(21)), (GTBO(1),VALUE(22)),
8     (CPO(1),VALUE(23))
```

then add the new node:

```
      NODET=NODPLC(LOC+36)
```

Set the power to zero:

```
      CP=0.0D0
```

In the DC model parameters section some additional variables need to be defined:

```
      PC=VALUE(LOCM+30)
      XTB=VALUE(LOCM+41)
      EG=VALUE(LOCM+42)
      XTI=VALUE(LOCM+43)
      VTT=VT (so Vt can vary with each device's temperature)
      TNOM=VALUE(ITEMPS+ITEMNO)+CTOK
```

Note: The nominal temperature is referenced as VALUE(ITEMPS+ITEMNO)+CTOK in Kelvin. It is always the current ambient temperature whether it is the default temperature, .TEMP value, or TNOM value.

Initialize temperature flag (used to indicate wether device is in the proper region of operation for thermal iteration):

```
NOTMP=.FALSE.
```

Initialization section

After statement 25 (INITF = 2) insert:

```
VTMP=0.0D0
```

This initializes the temperature to the ambient.

If the device is off there is no power dissipated. After statement 40 (INITF = 3) insert:

```
VTMP=0.0D0
```

After statement 50 (INITF=4):

```
VTMP=VTMPO(LX0+LOCT)
IF (VBC.GT.0.0D0) NOTMP=.TRUE.
```

After statement 60 (INITF=5):

```
VTMP=VTMPO(LX1+LOCT)
IF (VBC.GT.0.0D0) NOTMP=.TRUE.
```

After statement 70 (INITF=6):

```
IF (VBC.GT.0.0D0) NOTMP=.TRUE.
VTMPO(LX0+LOCT)=VTMPO(LX1+LOCT)
VTMP=(1.0D0+XFACT)*VTMPO(LX1+LOCT) - XFACT*VTMPO(LX2+LOCT)
GTCO(LX0+LOCT)=GTCO(LX1+LOCT)
GTBO(LX0+LOCT)=GTBO(LX1+LOCT)
```

After statement 100 (INITF=1), normal iteration:

```
IF (VBC.GT.0.0D0) NOTMP=.TRUE.
VTMP=VALUE(LVNIM1+NODET))
```

and after statement 110:

```
DELVTMP=VTMP-VTMPO(LX0+LOCT)
CCHAT=CCO(LX0+LOCT)+(GMO(LX0+LOCT)+GOO(LX0+LOCT))*DELVBE
```

```
1    -(GOO(LX0+LOCT)+GMUO(LX0+LOCT))*DELVBC+GTCO(LX0+LOCT)*DELVTMP
     CBHAT=CBO(LX0+LOCT)+GPIO(LX0+LOCT)*DELVBE+GMUO(LX0+LOCT)*DELVBC
1    +GTBO(LX0+LOCT)*DELVTMP
```

If not forward active, set power to zero:

```
     IF (NOTMP) THEN
     CPHAT=0.0D0
     ELSE
     CPHAT=CPO(LX0+LOCT)+DABS(CCO(LX0+LOCT)*DELVCE)
1          +DABS(CBO(LX0+LOCT)*DELVBE)
     ENDIF
```

Add a convergence check for the temperature in the last position of the BYPASS IF SOLUTION HAS NOT CHANGED section:

```
     TOL=RELTOL*DMAX1(DABS(VTMP),DABS(VTMPO(LX0+LOCT)))+VNTOL*1D2
     IF (DABS(DELVTMP).GE.TOL) GO TO 200
```

This avoids premature convergence in dc sweeps before the temperature has settled. After this add the lines:

```
     VTMP=VTMPO(LX0+LOCT)
     GTC=GTCO(LX0+LOCT)
     GTB=GTBO(LX0+LOCT)
     CP=CPO(LX0+LOCT)
```

Update of temperature-dependent variables

Temperature-dependent model parameters IS, ISE, ISC, BF, BR, EG, and VT are recomputed to reflect the local temperature increase. The following statements were added after the LIMIT NONLINEAR BRANCH VOLTAGES section:

```
210       CONTINUE
          IF (VTMP.LT.0.0D0) THEN
               VTMP=0.0D0
               NOTMP=.TRUE.
          ELSE
               LOOPTMP=.TRUE.
               DELTMP=1D-1
               ODELTMP=1D1
               VTMP=VTMP+DELTMP
          ENDIF
220       CONTINUE
```

```
DTEMP=VTMP/TNOM
RATIO=DTEMP+1.0D0
RATLOG=DLOG(RATIO)
TLOC=TNOM+VTMP
 VTT=BOLTZ*(TLOC)/CHARGE
 EG=EG0-(EGAP*TLOC*TLOC)/(TLOC+TGAP)
FACTLN=DTEMP*EG/VTT+XTI*RATLOG
FACTOR=DEXP(FACTLN)
CSAT=VALUE(LOCM+1)*AREA*FACTOR
BFACTR=DEXP(XTB*RATLOG)
BFM=VALUE(LOCM+2)*BFACTR
BRM=VALUE(LOCM+8)*BFACTR
C2=VALUE(LOCM+6)*AREA*DEXP(FACTLN/VALUE(LOCM+7))/BFACTR
C4=VALUE(LOCM+12)*AREA*DEXP(FACTLN/VALUE(LOCM+13))/BFACTR
```

These equations are equivalent to those in the temperature update routine, TMPUPD. Note: VTT is the new thermal voltage, and VT should be replaced by it throughout this subroutine.

Computation of the new conductance and current terms

The thermal model is valid only in the forward-active region, so should only be computed in this case. If the device is not in the forward-active region then temperature coefficients will be zero. The thermal transconductance is computed in the DETERMINE DC INCREMENTAL CONDUCTANCES section before statement 440:

```
IF (NOTMP) THEN
      CP=0.0D0
      GTB=0.0D0
      GTC=0.0D0
ELSEIF (LOOPTMP) THEN
      CPHI=DABS(VCE*CC)+DABS(VBE*CB)
      CCHI=CC
      CBHI=CB
      CC=0.0D0
      LOOPTMP=.FALSE.
      VTMP=VTMP-DELTMP
      GOTO 220
ELSE
      CP=DABS(VCE*CC)+DABS(VBE*CB)
      GTC=(CCHI-CC)*ODELTMP
      GTB=(CBHI-CB)*ODELTMP
```

```
      ENDIF
```

Check convergence

After statement 700, add a convergence check for power:

```
      TOL=RELTOL*1.0D3*DMAX1(DABS(CPHAT),DABS(CP))+ABSTOL
      IF (DABS(CPHAT-CP).LE.TOL) GO TO 780
```

Load the matrix

Save the state vector information:

```
780   ...
      VTMPO(LX0+LOCT)=VTMP
      GTCO(LX0+LOCT)=GTC
      GTBO(LX0+LOCT)=GTB
      CPO(LX0+LOCT)=CP
```

Compute and store the currents:

```
900   CEQBE=TYPE*(CC+CB-VBE*(GM+GO+GPI)+VBC*(GO-GEQCB)
     1        -VTMP*(GTC+GTB))
      CEQBC=TYPE*(-CC+VBE*(GM+GO)-VBC*(GMU+GO)+VTMP*GTC)
... (to the end of section)
      IF (.NOT.NOTMP) THEN
      VALUE(LVN+NODET)=VALUE(LVN+NODET)+(CP-VCE*(VCE*GO+VBE*GM
     1  +VTMP*GTC*TYPE)-VBE*(VBE*GPI+VTMP*GTB*TYPE))
      ENDIF
```

Stuff the conductance matrix:

```
950   ... (to the end of normal load)
      LOCY=LVN+NODPLC(LOC+37)
      VALUE(LOCY)=VALUE(LOCY)+GTC*TYPE
      LOCY=LVN+NODPLC(LOC+38)
      VALUE(LOCY)=VALUE(LOCY)+GTB*TYPE
      LOCY=LVN+NODPLC(LOC+39)
      VALUE(LOCY)=VALUE(LOCY) - (GTC+GTB)*TYPE
      LOCY=LVN+NODPLC(LOC+40)
      VALUE(LOCY)=VALUE(LOCY)-VCE*GO*TYPE
      LOCY=LVN+NODPLC(LOC+41)
      VALUE(LOCY)=VALUE(LOCY)-(VCE*GM+VBE*GPI)*TYPE
      LOCY=LVN+NODPLC(LOC+42)
      VALUE(LOCY)=VALUE(LOCY)+(VCE*(GM+GO)+VBE*GPI)*TYPE
      LOCY=LVN+NODPLC(LOC+43)
      VALUE(LOCY)=VALUE(LOCY)-(GTC*VCE+GTB*VBE)*TYPE
1000  RETURN
      END
```

Output of operating-point information

DCOP

This routine prints out operating-point information. The following changes allow thermal resistance, local temperature increase, and thermal conductance to be printed in response to the .OP command.

```
      DIMENSION
     2  CCS(12), CBX(12), RX(12),DTMP(12), GTC(12), GTB(12)

      DATA ADTMP, GTMP / 5HDEL_T, 3HGTC, 4HGTB /
```

Change BJT section:

In statement 320 and in the statement after 360 change:

NODPLC(LOC+36) → NODPLC(LOC+44)

then add:

```
...
      BETADC(KNTR)=CC(KNTR)/DSIGN(DMAX1(DABS(CB(KNTR)),1.0D-20),
     1        CB(KNTR))
      DTMP(KNTR)=VALUE(LOCT+19)
      GTC(KNTR)=VALUE(LOCT+20)
      GTB(KNTR)=VALUE(LOCT+21)
      IF (MODEDC.NE.1) GO TO 325
...
      IF (VALUE(LOCT+19) .EQ. 0.0D0) GO TO 355
      WRITE (IOFILE,AFMT2) ADTMP,(DTMP(I),I=1,KNTR)
      WRITE (IOFILE,AFMT2) AGTC,(GTC(I),I=1,KNTR)
      WRITE (IOFILE,AFMT2) AGTB,(GTB(I),I=1,KNTR)
355   IF (MODEDC.NE.1) GO TO 360
```

These thermal variables are printed only if a thermal model is used.

Changes for AC analysis

Changes to the ac analysis simulation code are similar to the corrections to the change in the dc analysis.

ACLOAD

In the BJT section, change the end of linked-list check:

```
260     IF ((LOC.EQ.0).OR.(NODPLC(LOC+44).NE.0)) GO TO 300
```

After the other admittance terms have been loaded the following code is added for the thermal model:

```
IF(NODPLC(LOC+36).NE.1) THEN
GTC=VALUE(LOCT+20)
GTB=VALUE(LOCT+21)
VBE=VALUE(LOCT)
VCE=VALUE(LOCT)-VALUE(LOCT+1)
CC=VALUE(LOCT+2)
LOCY=LYNL+NODPLC(LOC+37)
VALUE(LOCY)=VALUE(LOCY)+GTC
LOCY=LYNL+NODPLC(LOC+38)
VALUE(LOCY)=VALUE(LOCY)+GTB
LOCY=LYNL+NODPLC(LOC+39)
VALUE(LOCY)=VALUE(LOCY)-(GTC+GTB)
LOCY=LYNL+NODPLC(LOC+40)
VALUE(LOCY)=VALUE(LOCY)-VCE*(GO+GMU)-CC
LOCY=LYNL+NODPLC(LOC+41)
VALUE(LOCY)=VALUE(LOCY)-VCE*(GM-GMU)-VBE*GPI-CB
LOCY=LYNL+NODPLC(LOC+42)
VALUE(LOCY)=VALUE(LOCY)+VCE*(GM+GO)+VBE*GPI+CC+CB
LOCY=LYNL+NODPLC(LOC+43)
VALUE(LOCY)=VALUE(LOCY)-GTC*VCE-GTB*VBE
ENDIF
```

Table B-2 Offsets for BJT matrix elements.

LOC	C	B	E	c	b	e	s	T_{ext}
C	24			10				
B		25		34	11			
E			26			12		
c	13	35		27	14	15	32	37
b		16		17	28	18		38
e			19	20	21	29		39
s				33			31	
T_{ext}				40	41	42		43

APPENDIX C
SOISPICE CODE CHANGES FOR MODELING OF SELF-HEATING USING THE EXPLICITE TEMPERATURE APPROACH

Introduction

A physical charge-based model for partially-depleted (PD) SOI MOSFETs, with self-heating, non-local impact-ionization current, parasitic BJT, and other optional features which can be important in short-channel devices has been implemented in SOI-SPICE-4. Similar changes were made to the fully-depleted (FD) SOI MOSFET model routines.

PD SOI MOSFET Model User Reference

The user specifies the use of the self-heating model by including the RTH parameter in the device card as explained in Table C-1.

ZXXXXX ND NGF NS NGB (NB) (BJT) MNAME L=exp W=exp [AD=exp] [AS=exp] [AB=exp] [NRD=exp] [NRS=exp] [NRB=exp] [RTH=exp] [CTH=exp] [off] [IC = Vds, Vgfs, Vgbs, Vbs]

Table C-1 PD SOI MOSFET Device Line Explanation

Name	Description
ND	Drain node
NGF	Front gate node
NS	Source node
NGB	Back gate node
NB	Optional body node
BJT	Parasitic BJT (on) flag
RTH	Thermal resistance
CTH	Thermal capacitance

PD SOI MOSFET Model Implementation

The following figure shows how current and voltage are used as duals to power dissipation and temperature. The power of the device is applied as a current that generates a voltage as it flows through the thermal impedance, RTH and CTH, that is equivalent to the change in temperature of the device. The increase in temperature is used to update the device model parameters each iteration. If the parameter RTH is not specified on the device line, the above thermal subcircuit is not generated, and the standard isothermal model is used. The changes in SPICE2G.6 files made to implement self-heating in the PD SOI MOSFET model are marked with "Cdtz" comments in the Fortran source code. The implementations required changes to 17 subroutines and the addition of one new routine, all described below.

Subroutines ADDELT and ALTER

In these routines the matrix size defined by LNOD(16) was increased by 12 for the new admittance matrix elements.

Subroutine FIND

This routine checks for repeated element names and allocates memory for unique elements. The matrix size defined by LNOD(16) was increased by 12 for the new admittance matrix elements. Also, the IFORCE argument value of two is added to bypass the repeated name check. This is needed to allocate memory for the thermal capacitance,

since all thermal capacitors have the same name. This is explained in the MODCHK section.

Subroutine READIN

This routine decodes the input deck. First L2NOD(16) is increased by 12 for thermal elements of the admittance matrix. The device line parameters RTH and CTH were added to the list.

Subroutine MODCHK

This routine processes the model parameters. In the NFD section where the parasitic resistances are added, check for RTH, add the temperature node to the matrix, and add the CTH capacitor if needed.

```
IF (VALUE(LOCV+25).NE.0.0D0) THEN
      NUMNOD=NUMNOD+1
      NODPLC(LOC+47)=NUMNOD
      VALUE(LOCV+25)=1.0D0/VALUE(LOCV+25)
      IF (VALUE(LOCV+26).NE.0.0D0) THEN
        CALL FIND(ACGATOR,2,LOCTH,2)
        LOCVTH=NODPLC(LOCTH+1)
        NODPLC(LOCTH+2)=NUMNOD
        NODPLC(LOCTH+3)=1
        VALUE(LOCVTH+1)=VALUE(LOCV+26)
        NODPLC(LOCTH+4)=1
      ENDIF
ELSE
      NODPLC(LOC+47)=1
ENDIF
```

All the thermal capacitors are added to the normal capacitor list with the name GATOR; this name was chosen arbitrarily. FIND is forced to bypass the name-checking by the 2 in the call (see above).

Subroutines MATPTR and MATLOC

These routines reserve memory for the linked list. The thermal matrix elements and state variables are added as shown in the linked list provided at the end of this document.

Subroutine NFDFET

This subroutine, by calling NFDID and NFDEC, performs the initialization, matrix-load operations, and voltage-iterate limiting, and determine the operating-point currents and charges and their respective derivatives for the PD SOI MOSFET device. The following is a description of the thermal changes relating to self-heating.

Make a new common block for thermal arguments (used for FD devices also):

```
COMMON /TMPARG/ GMTMP,GIITMP,GIRTMP,GITTMP,GPWRTMP,GPWRGF,GPWRDS,
&      GPWRGB,GPWRBY,GD,GS
```

Extend dimension and equivalence for the thermal state-variables. Get temperature node (this will be a flag for temperature modeling).

```
NTMP=NODPLC(LOC+47)
```

Variables VTMP (temperature change) and CPWR (power dissipation) are treated the same as any other voltages and currents. The temperature change is limited to positive values. Add thermal arguments to NFDID call for normal and inverse modes:

```
CALL NFDID(VGFS,VDS,VGBS,VBS,LOC,QD,QGF,QS,QGB,QBY,
&      ...
&      CBD,CBGB,CBS,CDGB,CSGB,CGBD,VTMP,CPWR,CPWRTMP,
&      CGFTMP,CDTMP,CGBTMP,CBYTMP,CSTMP)
```

In the transient analysis, add thermal parameters to NFDEC call and include in tranconductance computations for normal and inverse modes:

```
CALL NFDEC(VGFD,VGFS,VGFB,VGBD,VGBS,VGBB,XNQFF,XNQFB,
&     ...
&     CDTMP,CSTMP,CGFTMP,CGBTMP,CBYTMP,GCDTMP,GCSTMP,GCGFTMP,
&     GCGBTMP,GCBYTMP)
GCGFS=-(GCGFGF+GCGFD+GCGFB+GCGFTMP)
GCDS=-(GCDGF+GCDD+GCDB+GCDGB+GCDTMP)
GCBS=-(GCBGF+GCBD+GCBB+GCBGB+GCBYTMP)
GCGBS=-(GCGBB+GCGBD+GCGBGB+GCGBTMP)
GCSS=-(GCSGF+GCSD+GCSB+GCSGB+GCSTMP)
```

Conditionally compute current-vector element for power and add temperature parameters to other vector elements for normal and inverse modes, and load power element of current vector. Also check if drain and source resistances need updating:

```
IF (NTMP.NE.1) THEN
    VALUE(LVN+NTMP)=VALUE(LVN+NTMP)+CPWREQ
    IF (VTMP.GT.0.0D0) THEN
    GDPR=GD
    GSPR=GS
    ENDIF
  ENDIF
```

Load thermal elements of the admittance matrix.

Subroutine NFDID

This subroutine, by calling NFDMOD, numerically calculates the derivatives of the operating-point currents and charges for both the forward and inverse modes of operation of the NFD/SOI device. The same changes to the common block and subroutine arguments done for NFDFET are repeated here. An additional call to NFDMOD is made to find the temperature derivatives.

Subroutine NFDEC

This subroutine calculates the transcapacitance terms used in transient and AC analysis from the intrinsic and extrinsic components of charge in the NFD/SOI device. Thermal transient conductance terms are also computed in this routine.

Subroutine NFDMOD

This subroutine, the primary model routine, calculates the channel transport, impact-ionization, junction leakage, and if requested the parasitic bipolar transistor currents of the PD SOI MOSFET for the given internal device voltages, as well as the intrinsic charge components that define the charging currents. After the model parameters are loaded and before any computations are made, SOITMP is called to update the temperature dependent parameters for the local temperature. The local thermal voltage VTT is substituted everywhere the global thermal voltage VT was used.

It is important to assign the model parameter values to local variables before the SOITMP call and not to directly access the model parameter values (e.g. VALUE(LOCM+?)) in subsequent calculations. Otherwise these model parameters would not reflect changes in local temperature, which could cause inaccurate results.

Subroutine SOITMP

This subroutine updates all of the temperature dependent parameters for each device on each iteration. It is called by the NFDMOD routine. The parameters affected are shown in Table C-2.

Table C-2 Temperature dependent parameters.

NAME	Description
XNIN	Intrinsic carrier density
EGFET	Silicon energy gap
VTT	Local Thermal Voltage
XMULDSJ	LDD carrier mobility
XMUBH	Minority carrier mobility in high-doped body
UO	Zero field electron mobility
VSAT	Carrier saturated drift velocity
WKF	Front-gate work function difference
WKB	Back-gate work function difference
VFBF	Front-gate flatband voltage
VFBB	Back-gate flatband voltage
JRO	Body-source junction recombination current coefficient
ALPHA	Impact-ionization parameter a_o
TAUG	Carrier lifetime in lightly doped region
VBI	Source to bulk built-in potential
PHIB	Twice fermi potential of body
RLDD	LDD sheet resistivity
BETA	b
FSAT	fsat
GD	Drain parasitic resistance
GS	Source parasitic resistance

These are the same dependencies used in TMPUPD implemented by Glen Workman [Wor96]. If temperature dependencies are changed or added to TMPUPD, corresponding changes must also be made to SOITMP.

Subroutine DCOP

This routine generates the output for the operating point. If self-heating is modeled for a device, the change in temperature, power dissipated, and relevant conductance and capacitance terms are printed.

Subroutine ACLOAD

This routine loads the complex admittance matrix for the ac analysis. If there is a device with self-heating, the thermal elements of the matrix are loaded. See description of NFDFET routine.

Data Structures

The data structures for the PD SOI MOSFET device (ID=16) and model (ID=26), and for the print (ID=41) and plot (ID=42) are given in Table C-3 for SOI-SPICE-4.3i. All other linked lists remain the same as in the original SPICE2G.6.

Table C-3 Offsets for PD SOI MOSFET matrix elements.

LOC	D	Gf	S	Gb	B	d	s	b	T
D	38					13			
Gf		39				14	15	16	49
S			40				17		
Gb				41		18	19	20	50
B					42			21	
d	22	23		24		43	25	26	51
s		27	28	29		30	44	31	52
b		32		33	34	35	36	45	53
T		54		55		56	57	58	48

PD SOI MOSFET Transistor (ID=16)

	-01: subckt info		
LOC	+00: next pointer	LOCV	+00: device name
	+01: LOCV		+01: channel length L
	+02: nd		+02: channel width W
	+03: ngf		+03: IC - vds
	+04: ns		+04: IC - vgfs
	+05: ngb		+05: IC - vgbs
	+06: nb		+06: IC - vbs
	+07: ndp		+07: DEVMOD
	+08: nsp		+08: AD
	+09: nbp		+09: AS
	+10: model pointer		+10: AB
	+11: off		+11: GD (NRD)
	+12: parasitic BJT flag		+12: GS (NRS)
	+13: (nd,ndp)		+13: GB (NRB)
	+14: (ngf,ndp)		+14: BETA (MODCHK)
	+15: (ngf,nsp)		+15: FSAT (MODCHK)
	+16: (ngf,nbp)		+16: DICEFAC (MODCHK)
	+17: (ns,nsp)		+17: XLMODFAC (MODCHK)
	+18: (ngb,ndp)		+18: RLDD (MODCHK)
	+19: (ngb,nsp)		+19: COVLGFD
	+20: (ngb,nbp)		+20: COVLGFS
	+21: (nb,nbp)		+21: COVLGFB
	+22: (ndp,nd)		+22: <unused>
	+23: (ndp,ngf)		+23: <unused>
	+24: (ndp,ngb)		+24: COVLGBB
	+25: (ndp,nsp)		+25: RTH
	+26: (ndp,nbp)		+26: CTH
	+27: (nsp,ngf)		+27: <unused>
	+28: (nsp,ns)		+28: <unused>
	+29: (nsp,ngb)		+29: <unused>
	+30: (nsp,ndp)		
	+31: (nsp,nbp)		
	+32: (nbp,ngf)		
	+33: (nbp,ngb)		+47: nt
	+34: (nbp,nb)		+48: (nt,nt)
	+35: (nbp,ndp)		+49: (ngf,nt)
	+36: (nbp,nsp)		+50: (ngb,nt)
	+37: Lxl offset		+51: (ndp,nt)
	+38: (nd,nd)		+52: (nsp,nt)
	+39: (ngf,ngf)		+53: (nbp,nt)
	+40: (ns,ns)		+54: (nt,ngf)
	+41: (ngb,ngb)		+55: (nt,ngb)
	+42: (nb,nb)		+56: (nt,ndp)
	+43: (ndp,ndp)		+57: (nt,nsp)
	+44: (nsp,nsp)		+58: (nt,nbp)
	+45: (nbp,nbp)		
	+46: body mode (0 for float.)		

PD SOI MOSFET Model (ID=26)

-01:	subckt info		
LOC +00:	next pointer	LOCM +00:	model name
+01:	LOCM	+01:	VFBF
+02:	model type	+02:	VFBB
		+03:	TOXF
		+04:	TOXB
		+05:	WKF
		+06:	WKB
		+07:	NQFF
		+08:	NQFB
		+09:	NSUB
		+10:	NGATE
		+11:	FNK
		+12:	FNA
		+13:	TF
		+14:	TB
		+15:	THALO
		+16:	NBL
		+17:	NBH
		+18:	NHALO
		+19:	UO
		+20:	THETA
		+21:	VSAT
		+22:	TPG
		+23:	TPS
		+24:	NLDS
		+25:	NDS
		+26:	LLDS
		+27:	LLDD
		+28:	LDIFF
		+29:	ZETA
		+30:	ETA
		+31:	LMOD
		+32:	ALPHA
		+33:	BETA
		+34:	BFACT
		+35:	CGFDO
		+36:	CGFSO
		+37:	CGFBO
		+38:	RD
		+39:	RS
		+40:	RB
+46:	JRO	+41:	TAUO
+47:	M	+42:	DL
+48:	VJMIN	+43:	DW
+49:	<unavailable	+44:	RHOSD
+50:	FVBJT	+45:	RHOB

State Variable Table (PD SOI MOSFET Model)

Lxl +00:	vbso (VBS)
+01:	vbdo (VBD)
+02:	vgfso (VGFS)

+03: vgfdo (VGFD)
+04: vgbso (VGBS)
+05: vdso (VDS)
+06: cicho (ICH)
+07: ciro (IR)
+08: cigto (IGT)
+09: cigio (IGI)
+10: cbjto (IBJT)
+11: gidgfso (GDGF)
+12: giddso (GDD)
+13: gidbso (GDB)
+14: gligfso (GGIGF)
+15: glidso (GGID)
+16: glibso (GGIB)
+17: girbso GRB)
+18: qdo (QD)
+19: cqdo (IQD)
+20: qgfo (QGF)
+21: cqgfo (IQGF)
+22: qso (QS)
+23: cqso (iQS)
+24: qgbo (QGB)
+25: cqgbo (iQGB)
+26: qbo (QB)
+27: cqbo (iQB)
+28: cdgfo
+29: cddo
+30: cdbo
+31: cdso
+32: cgfgfo
+33: cgfdo
+34: cgfbo
+35: cgfso
+36: csgfo
+37: csdo
+38: csbo
+39: csso
+40: cgbgbo
+41: cgbbo
+42: cgbso
+43: cbgfo
+44: cbdo
+45: cbgbo
+46: cbbo
+47: cbso
+48: vtso
+49: vtwo
+50: vdsato
+51: vdsato
+52: leffo
+53: cgbdo
+54: cdgbo
+55: csgbo
+56: qno
+57: vtmpo (temperature)
+58: cpwro (power)
+59: gmtmpo
+60: gittmpo
+61: girtmpo
+62: cgttmpo
+63: cdtmpo
+64: cstmpo
+65: cgbtmpo
+66: cbytmpo
+67: gpwrtmpo
+68: gpwrgfo
+69: gpwrdso
+70: gpwrgbo
+71: gpwrbyo

Comments (PD SOI MOSFET Model):

"Model Type" is +1 for NNFDSOI and -1 for PNFDSOI.

The following preprocessing is performed in subroutine MODCHK:

Location:	Stored Term:

LOCM+NZPARAM+1	VBi (S-B built-in potential)
LOCM+NZPARAM+2	CB=EPSSIL/TB
LOCM+NZPARAM+3	COXF=EPSOX/TOXF
LOCM+NZPARAM+4	COXB=EPSOX/TOXB
LOCM+NZPARAM+5	QB=-CHARGE*TB*NBL
LOCM+NZPARAM+6	PHIB=2*VT*DLOG(NBL/Ni)
LOCM+NZPARAM+7	TLDD=TF-THALO
LOCM+NZPARAM+8	XALPHA=1+CB/COXF
LOCM+NZPARAM+9	XLC=TB(CB/(2*COXF*XALPHA))$^{1/2}$
LOCM+NZPARAM+10	<unused>
LOCM+NZPARAM+11	<unused>
LOCM+NZPARAM+12	<unused>
LOCM+NZPARAM+13	XMUBH (minority-carrier mobility in high-doped body)
LOCM+NZPARAM+14	<unused>
LOCM+NZPARAM+15	(unused>
LOCM+NZPARAM+16	ESLOPE=CHARGE*NLDS/EPSSiL
LOCM+NZPARAM+17	TAUG=2*TAUO/(1+NBL/NO)
LOCM+NZPARAM+18	XMOBFAC1=THETA*COXF/(2*EPSSIL)
LOCM+NZPARAM+19	XBFACT=BFACT*XMOBFAC1/ (2-XALPHA)

Note: NZPARAM=50 for current number of input parameters.

Additional COMMON blocks:

COMMON /NFDARG/ CICH,CIR,CIGT,CIGI,CBJT,GIDGFS,GIDDS,GIDBS, GIIGFS,GIIDS,GIIBS,GIRBS,VTS,VTW,XLEFF,VDSAT

COMMON /TMPARG/ GMTMP,GIITMP,GIRTMP,GITTMP,GPWRTMP, GPWRGF,GPWRDS,GPWRGB,GPWRBY

Output Variables (ID=41,42)

-01:	subckt info		
LOC +00:	next pointer	LOCV +00:	Element Name
+01:	LOCV	+01:	Device Name
+02:	Device ID / LOC	+02:	Oper Var Name
+03:	KOFF (LOCT/LOCV offset)		
+04:	ISEQ		
+05:	type=9		
+06:	KLOC (KLOCT/KLOCV offset)		
+07:	ISTATE (S=0, P=1)		

Comments:
For ID=41,42 and type (NODPLC(LOC+5)) = 0, 1, and 2-8, the definitions of the variables remain as defined in the "Program Reference for SPICE2".

APPENDIX D
SPICE CIRCUIT DESCRIPTIONS USED FOR SIMULATION EXAMPLES

Single BJT Common-Emitter Amplifier

This is a single NPN common-emitter amplifier circuit with an open-circuit load used to generate Figure 4-1.. Dc operating point and ac analyses are executed.

```
COMMON-EMITTER AMPLIFIER CIRCUIT
VBE 2 0 .82 AC 1
VCE 1 0 5
LT 1 3 1G
Q1 3 2 0 0 4H3X50N
XZTH 4 0 ZTH
*Three-pole Model
.SUBCKT ZTH 1 4
RTH1 1 2 92.4
CTH1 1 2 1.02u
RTH2 2 3 142.8
CTH2 2 3 0.187u
RTH3 3 4 44.8
CTH3 3 4 55.6n
.ENDS
*
.MODEL H3X50N NPN (IS=1.875E-16 XTI=3 EG=1.16 VAF=60 VAR=4.5
+ BF=146.1 ISE=1.875E-19 NE=1.4 IKF=7.5E-2 XTB=2 BR=10
+ ISC=1.875E-14 NC=1.8 IKR=7.5E-2 RC=29.4 CJC=2.818E-13
+ MJC=0.24 VJC=0.97 FC=0.5 CJE=3.9E-13 MJE=0.51 VJE=0.872
+ TR=4E-9 TF=17.85E-12 ITF=1.155 XTF=78.81 VTF=10 PTF=0
+ XCJC=0.1756 CJS=1.689E-13 VJS=0.75 MJS=0 RE=1.333 RB=35.18
+ RBM=0 KF=0 AF=1 EGAP=7.02E-4 TGAP=1108)
*Simulation Section
.OPTIONS NUMDGT=7 LIMPTS=2000 RELTOL=1e-5 ITL1=1000 ITL2=1000
.WIDTH OUT=80
.OP
.AC DEC 5 1 1G
.PRINT AC V(3)
.END
```

741 Operational Amplifier Circuit

This is the 741 operational amplifier found in Grey and Meyer [Gra84]. The schematic in shown in Figure 4-2, and the simulation results and shown in Figures 4-3 and 4-4. Dc operating point, dc transfer and ac analyses are executed.

```
SELF-HEATING 741 OP-AMP CIRCUIT
VCC 1 0 DC 15
VEE 6 0 DC -15
VI1 8 0 DC 0
VI2 9 0 DC .835M AC 1
R1 15 6 1K
R2 16 6 1K
R3 14 6 50K
R4 5 6 5K
R5 2 3 39K
R6 24 25 27
R7 25 26 22
R8 20 6 100
R9 19 6 50K
R10 22 23 40K
CC 13 17 .65P
Q1 7 8 10 6 51 H3x50N
Q2 7 9 11 6 52 H3x50N
Q3 12 4 10 1 53 H3x50P
Q4 13 4 11 1 54 H3x50P
Q5 12 14 15 6 55 H3x50N
Q6 13 14 16 6 56 H3x50N
Q7 1 12 14 6 57 H3x50N
Q8 7 7 1 1 58 H3x50P
Q9 4 7 1 1 59 H3x50P
Q10 4 3 5 6 60 H3x50N
Q11 3 3 6 6 61 H3x50N
Q12 2 2 1 1 62 H3x50P
Q13A 21 2 1 1 63 H3x50P .25
Q13B 17 2 1 1 64 H3x50P .75
Q14 1 21 24 6 65 H3x50N
Q16 1 13 19 6 66 H3x50N
Q17 17 19 20 6 67 H3x50N
Q18 21 22 23 6 68 H3x50N
Q19 21 21 22 6 69 H3x50N
Q20 6 23 26 1 70 H3x50P
Q23A 6 17 23 1 73 H3x50P
XZTH1 51 0 ZTH
XZTH2 52 0 ZTH
XZTH3 53 0 ZTH
XZTH4 54 0 ZTH
```

741 op-amp circuit continued.

```
XZTH5 55 0 ZTH
XZTH6 56 0 ZTH
XZTH7 57 0 ZTH
XZTH8 58 0 ZTH
XZTH9 59 0 ZTH
XZTH10 60 0 ZTH
XZTH11 61 0 ZTH
XZTH12 62 0 ZTH
XZTH13 63 0 ZTH
XZTH14 64 0 ZTH
XZTH15 65 0 ZTH
XZTH16 66 0 ZTH
XZTH17 67 0 ZTH
XZTH18 68 0 ZTH
XZTH19 69 0 ZTH
XZTH20 70 0 ZTH
XZTH23 73 0 ZTH
*Three-pole Model
.SUBCKT ZTH 1 4
RTH1 1 2 92.4
CTH1 1 2 1.02u
RTH2 2 3 142.8
CTH2 2 3 0.187u
RTH3 3 4 44.8
CTH3 3 4 55.6n
.ENDS
*
.MODEL H3X50N NPN (IS=1.875E-16 XTI=3 EG=1.16 VAF=60 VAR=4.5
+ BF=146.1 ISE=1.875E-19 NE=1.4 IKF=7.5E-2 XTB=2 BR=10
+ ISC=1.875E-14 NC=1.8 IKR=7.5E-2 RC=29.4 CJC=2.818E-13
+ MJC=0.24 VJC=0.97 FC=0.5 CJE=3.9E-13 MJE=0.51 VJE=0.872
+ TR=4E-9 TF=17.85E-12 ITF=1.155 XTF=78.81 VTF=10 PTF=0
+ XCJC=0.1756 CJS=1.689E-13 VJS=0.75 MJS=0 RE=1.333 RB=35.18
+ RBM=0 KF=0 AF=1 EGAP=7.02E-4 TGAP=1108)
*
.MODEL H3X50P PNP (IS=1.02E-16 XTI=3 EG=1.16 VAF=30 VAR=4.5
+ BF=70.11 ISE=1.02E-19 NE=1.4 IKF=7.5E-2 XTB=2 BR=7
+ ISC=1.02E-14 NC=1.8 IKR=7.5E-2 RC=38 CJC=4.27E-13
+ MJC=0.3 VJC=1.23 FC=0.5 CJE=4.8E-13 MJE=0.57 VJE=0.88
+ TR=4E-9 TF=33.91E-12 ITF=0.7127 XTF=45.14 VTF=10 PTF=0
+ XCJC=0.1756 CJS=1.689E-13 VJS=0.75 MJS=0 RE=1.333 RB=37.4
+ RBM=0 KF=0 AF=1 EGAP=7.02E-4 TGAP=1108)
*Simulation Section
.OPTIONS NUMDGT=7 TNOM=25 ACCT
.OP
.DC VI2 0 3m 10u
.AC DEC 5 1 100meg
.PRINT DC V(25)
.PRINT AC VM(25)
.END
```

Pin-Driver Transient

This is the pin-driver circuit to demonstrate the transient effects of self-heating. The schematic for this circuit is shown in Figure 4-5, and the results are shown in Figures 4-6 and 4-7.

```
*THERMAL PIN-DRIVER CIRCUIT
VIN 3 2 5 pulse (2 18 1u 5n 5n 500u 1000u)
Q1 2 3 4 1 10 H3X50P
Q2 7 7 6 1 11 H3X50P
Q3 5 5 4 2 12 H3X50N
Q4 1 5 6 2 13 H3X50N
I1 1 5 DC 1m
I2 7 2 DC 10m
VEE 2 0 DC -10
VCC 1 0 DC 10
XZTH1 10 0 ZTH
XZTH2 11 0 ZTH
XZTH3 12 0 ZTH
XZTH4 13 0 ZTH
*Three-pole Model
.SUBCKT ZTH 1 4
RTH1 1 2 92.4
CTH1 1 2 1.02u
RTH2 2 3 142.8
CTH2 2 3 0.187u
RTH3 3 4 44.8
CTH3 3 4 55.6n
.ENDS
*
.MODEL H3X50N NPN (IS=1.875E-16 XTI=3 EG=1.16 VAF=60 VAR=4.5
+ BF=146.1 ISE=1.875E-19 NE=1.4 IKF=7.5E-2 XTB=2 BR=10
+ ISC=1.875E-14 NC=1.8 IKR=7.5E-2 RC=29.4 CJC=2.818E-13
+ MJC=0.24 VJC=0.97 FC=0.5 CJE=3.9E-13 MJE=0.51 VJE=0.872
+ TR=4E-9 TF=17.85E-12 ITF=1.155 XTF=78.81 VTF=10 PTF=0
+ XCJC=0.1756 CJS=1.689E-13 VJS=0.75 MJS=0 RE=0 RB=35.18
+ RBM=0 KF=0 AF=1 EGAP=7.02E-4 TGAP=1108)
*
.MODEL H3X50P PNP (IS=1.02E-16 XTI=3 EG=1.16 VAF=30 VAR=4.5
+ BF=70.11 ISE=1.02E-19 NE=1.4 IKF=7.5E-2 XTB=2 BR=7
+ ISC=1.02E-14 NC=1.8 IKR=7.5E-2 RC=38 CJC=4.27E-13
+ MJC=0.3 VJC=1.23 FC=0.5 CJE=4.8E-13 MJE=0.57 VJE=0.88
+ TR=4E-9 TF=33.91E-12 ITF=0.7127 XTF=45.14 VTF=10 PTF=0
+ XCJC=0.1756 CJS=1.689E-13 VJS=0.75 MJS=0 RE=0 RB=37.4
+ RBM=0 KF=0 AF=1 EGAP=7.02E-4 TGAP=1108)
*Simulation Section
.OPTIONS LIMPTS=2000 RELTOL=1E-4 METHOD=GEAR
.TRAN 1U 250U
.PRINT TRAN V(3) V(7) V(10) V(13)
.END
```

Thermal Coupling of Three-Finger Emitter

This is the three-finger emitter circuit to demonstrate the thermal coupling effects that are now possible to simulate. The schematic for this circuit is shown in Figure 4-8, and the results are shown in Figure 4-9.

```
THREE FINGER NPN UHF 3X50UM DC TEST
IB 0 1 .15M AC 1
VCE 2 0 5
QNPN1 2 1 0 0 4 H3X50N
QNPN2 2 1 0 0 8 H3X50N
QNPN3 2 1 0 0 12 H3X50N
RTH1 4 5 280
HTH12 5 6 VTH2 100
HTH13 6 7 VTH3 20
VTH1 7 0 0
RTH2 8 9 280
HTH23 9 10 VTH1 100
HTH21 10 11 VTH3 100
VTH2 11 0 0
RTH3 12 13 280
HTH31 13 14 VTH1 20
HTH32 14 15 VTH2 100
VTH3 15 0 0
*
.MODEL H3X50N NPN (IS=1.875E-16 XTI=3 EG=1.16 VAF=60 VAR=4.5
+ BF=146.1 ISE=1.875E-19 NE=1.4 IKF=7.5E-2 XTB=2 BR=10
+ ISC=1.875E-14 NC=1.8 IKR=7.5E-2 RC=29.4 CJC=2.818E-13
+ MJC=0.24 VJC=0.97 FC=0.5 CJE=3.9E-13 MJE=0.51 VJE=0.872
+ TR=4E-9 TF=17.85E-12 ITF=1.155 XTF=78.81 VTF=10 PTF=0
+ XCJC=0.1756 CJS=1.689E-13 VJS=0.75 MJS=0 RE=1.333 RB=35.18
+ RBM=0 KF=0 AF=1 EGAP=7.02E-4 TGAP=1108)
*Simulation Section
.OPTIONS NUMDGT=7 LIMPTS=2000 RELTOL=1e-5 ITL1=1000 ITL2=1000
.WIDTH OUT=80
.DC VCE .1 10 0.05
.PRINT DC &(QNPN1,IC) &(QNPN2,IC)
.END
```

SOI MOSFET Characteristic Curves

This is a single n-channel partially-depleted SOI MOSFET circuit for generating the characteristic curves with and without self-heating. The results for this circuit are shown in Figure 6-5.

```
PD SOIMOSFET characteristic curves with self-heating
*voltage source
VCC 5 0 DC 2.5
VGB1 3 0 0
VGF1 11 0 DC 2
*current detection
VDUM1 5 21 0
VDUM2 5 22 0
ZN1 21 11 0 3  BJT MODN L=0.2E-6 W=10E-6 AD=10P AS=10P RTH=K CTH=5P
ZN2 22 11 0 3  BJT MODN L=0.2E-6 W=10E-6 AD=10P AS=10P
*
.MODEL MODN NNFDSOI (insert model here)
*Simulation Section
.DC VCC 0 3 0.1 VGF1 1 2 .5
.PRINT DC  I(VDUM1) I(VDUM2)
.OPTION LIMPTS=2000 METHOD=GEAR ITL1=500 ITL2=200 RELTOL=1E-5
.END
```

SOI MOSFET Step Response

This is a single n-channel partially-depleted SOI MOSFET circuit for generating the transient step-response with and without self-heating. The results and schematic for this circuit are shown in Figure 6-6.

```
PD SOIMOSFET step response
*Voltage source
VDD 5 0 2.5
VGF 11 0 PULSE (2 3 10N 10N 10N 50N 120N)
VGB 4 0 0
*first stage:
ZN1 5 11 0 4 0 BJT MODN L=0.2E-6 W=10E-6 AD=10P AS=10P RTH=8K CTH=115P
ZN2 5 11 0 4 0 BJT MODN L=0.2E-6 W=10E-6 AD=10P AS=10P
*
.MODEL MODN NNFDSOI (insert model here)
*Simulation Section
.TRAN 10N 5U
*.PRINT TRAN V(21) V(22)
*.PRINT TRAN V(22)
.PRINT TRAN &(ZN1,TEMP) V(11)
.OPTION LIMPTS=2000 METHOD=GEAR ITL1=500 ITL2=200 RELTOL=1e-4
.END
```

SOI MOSFET Inverter Switching

This is a complementary partially-depleted SOI MOSFET inverter circuit for generating the transient switching waveforms with and without self-heating. The schematic for the circuit is shown in Figure 6-7, and the results are shown in Figure 6-8.

```
PD SOI MOSFET Inverter circuit
VON 1 0 PULSE 0 3 2N  5N 5N 50N 100N
VDD 9 0 3
VGB1 20 0 0
*Inverter
* without RTH and CTH
*ZN0 2 1 0 20 BJT MODN L=.2E-6 W=11.6E-6 AD=7.54P AS=12.6P IC=0,0,0,0.9
*ZP0 2 1 9 20 BJT MODP L=.2E-6 W=15.7E-6 AD=10.2P AS=23.6P IC=0,0,0,-0.9
* with RTH and CTH
ZN0 2 1 0 20 BJT MODN L=.2E-6 W=11.6E-6 AD=7.54P AS=12.6P
+ RTH=7.1431E+03 CTH= 9.7547E-11   IC=0,0,0,0.9
ZP0 2 1 9 20 BJT MODP L=.2E-6 W=15.7E-6 AD=10.2P AS=23.6P
+ RTH=5.7134E+03 CTH= 1.3202E-10   IC=0,0,0,-0.9
CLOAD 2 0 5p
*
.MODEL MODN NNFDSOI (insert model here)
.MODEL MODP PNFDSOI (insert model here)
*Simulation Section
.TRAN 10N 2U
.PRINT  TRAN V(2) &(ZN0,TEMP)
.OPTION  ACCT NODE OPTS LIST LIMTIM=50 METHOD=GEAR
+       PIVTOL=1E-30
+       GMIN=1.00E-16 RELTOL=1.00E-03 ABSTOL=1.00E-12
+       VNTOL=1.00E-06 TRTOL=7.00E+00
+       ITL1=1.00E+02 ITL2=1.00E+02 ITL3=4.00E+00 ITL4=2.50E+02
+       ITL5=0.00E+00 LVLTIM=2.00E+00 TNOM=+2.50E+01 LIMPTS=99999
+       CHGTOL=1.00E-20
.END
```

BIBLIOGRAPHY

[Bro93] J. S. Brodsky, D. T. Zweidinger, and R. M. Fox, "Physics-based multiple-pole models for BJT self-heating", Proc. IEEE Bipolar Circuits and Technology Meeting, pp. 249-252, 1993.

[Bro97a] J. S. Brodsky, R. M. Fox, D. T. Zweidinger, and S. Veeraraghavan, "Physics-based dynamic self-heating model for SOI MOSFETs," to be published in IEEE Transactions on Electron Devices, vol. 44, no. 6, June 1997.

[Bro97b] J. S. Brodsky, Physics-based thermal impedance model for the simulation of self-heating in semiconductor devices and circuits, Ph.D. Dissertation, University of Florida, Gainesville, 1997.

[Dav92] C. Davis, G. Bajor, J. Butler, T. Crandell, J. Delgado, T. Jung, Y. Khajeh-Noori, B. Lomenick, V. Milam, H. Nicolay, S. Richmond and T. Rivoli, "UHF-1: A high speed complementary bipolar analog process on SOI", Proc. IEEE Bipolar Circuits and Technology Meeting, pp. 260-261, 1992.

[Dav88] W. F. Davis and M. L. Lidke, "The effect of thermal feedback within the bipolar transistor on Gummel-Poon model accuracy," Motorola Internal Report, May 27, 1988.

[Den89] R. T. Dennison and K. M. Walter, "Local thermal effects in high performance bipolar circuits/devices," Proc. IEEE Bipolar Circuits and Technology Meeting, pp. 164-167, 1989.

[Ebe54] J. J. Ebers and J. L. Moll, "Large-signal behavior of junction transistors," Proc. IRE, vol. 42, pp. 1761-1772, Dec. 1954.

[Fos96] J. G. Fossum, R. M. Fox and D. T. Zweidinger, "SOISPICE-4 (Ver. 4.4) Silicon-On-Insulator MOSFET SPICE with Self-Heating," Programmer's Reference Manual, University of Florida Department of Electrical and Computer Engineering, Oct. 1996.

[Fox91a] R. M. Fox and S.-G. Lee, "Predictive modeling of themal effects in BJTs," Proc. IEEE Bipolar Circuits and Technology Meeting, pp. 89-92, Sept. 1991.

[Fox91b] R. M. Fox and S.-G. Lee, "Scalable small-signal model for BJT self-heating," IEEE Electron Device Letters, vol. 12, no. 12, pp. 649-651, Dec. 1991.

[Fox91c] R. M. Fox and S.-G. Lee, "Thermal parameter extraction for bipolar circuit modeling", Electronics Letters, vol. 27, no. 19, pp. 1719-1720, 1991.

[Fox93a] R. M. Fox and J. S. Brodsky, "Effects of self-heating-induced negative output conductance in SOI circuits," SOI Conf. '93, Oct. 1993,

[Fox93b] R. M. Fox, S.-G. Lee, and D. T. Zweidinger, "The effects of BJT self-heating on circuit behavior", IEEE J. Solid-State Circuits, vol. 28, no. 6, pp. 678-685, June 1993.

[Fuk76] K. Fukahori and P. R. Gray, "Computer simulation of integrated circuits in the presence of electrothermal interaction," IEEE J. Solid-State Circuits, vol. SC-11, pp. 834-846, Dec. 1976.

[Gan92] P. Ganci, et al., "Self-heating in high-performance bipolar transistors fabricated on SOI substrates," Technical Digest IEEE International Electron Devices Meeting, pp. 417-420, 1992.

[Gao89] G.-B. Gao, M.-Z. Wang, X. Gui, and H. Morkoç, "Thermal design studies of high-power heterojunction bipolar transistors", IEEE Trans. Electron Devices, vol. 36, no. 5, pp. 854-863, May 1989.

[Get78] I. E. Getreu, Modeling the Bipolar Transistor, Elsevier Scientific, New York, 1978.

[Goo95] K. E. Goodson, M. I. Flik, L. T. Su and D. A. Antoniadis, "Prediction and Measurement of Temperature Fields in Silicon-On-Insulator Electronic Circuits," ASME Journal of Heat Transfer, vol. 117, pp. 574-581, August 1995.

[Gra84] P. R. Gray and R. G. Meyer, Analysis and Design of Analog Integrated Circuits, Third Edition, John Wiley & Sons, Inc., New York, 1993.

[Gum70] H. K. Gummel and H. C. Poon, "An integral charge control model of bipolar transistors", The Bell System Technical Journal, pp. 827-851, May-June 1970.

[Joy70] R. C. Joy and E. S. Schlig, "Thermal properties of very fast transistors," IEEE Trans. Electron Devices, vol. ED-17, no. 8, pp. 586-594, Aug. 1970.

[Lat81] M. Latif and P. R. Bryant, "Multiple equilibrium points and their significance in the second breakdown of bipolar transistors", IEEE J. Solid-State Circuits, vol. SC-16, pp. 8-15, Feb. 1981.

[Lee89] C. C. Lee, A. L. Palisoc, and Y. J. Min, "Thermal analysis of integrated circuit devices and packages,"IEEE Trans. on Components, Hybrids, and Manufacturing Technology, vol. 12, pp. 701-709, Dec. 1989.

[Lee92] S.-G. Lee, Predictive modeling of high-current output resistance and thermal effects in bipolar junction transistors. Ph.D dissertation, University of Florida, Gainesville FL, Aug. 1992.

[Lee95] T. -Y. Lee and R. M. Fox, "Extraction of thermal resistance for fully-depleted SOI MOSFET's," in Proceedings IEEE International SOI Conference, pp. 78-79, October 1995.

[Lia94] M. Liang and M. E. Law, "Influence of lattice self-heating and hot-carrier transport on device performance," IEEE Trans. Electron Devices, vol. 41, no. 12, pp. 2391-2398, Dec. 1994.

[May67] P. D. Maycock, "Thermal conductivity of silicon, germanium, III-V compounds, and III-V alloys," Solid-State Electronics, vol.10, pp. 161-168, 1967.

[McA92] C. C. McAndrew, "A complete and consistent electrical/ thermal HBT model," Proc. IEEE Bipolar Circuits and Technology, pp. 200-203, Oct. 1992.

[Mei77] G. Meijer, "The current dependency of the output conductance of voltage-driven bipolar transistors," IEEE J. Solid-State Circuits, pp. 428-429, Aug. 1977.

[Mun91] P. C. Munro and F.-Q. Ye, "Simulating the current mirror with a self-heating BJT model," IEEE J. Solid-State Circuits, pp. 1321-1324, Sept. 1991.

[Mül64] O. Müller, "Internal thermal feedback in four-poles especially in transistors," Proc. IEEE, vol. 52, pp. 924-930, Aug. 1964.

[Mül70] O. Müller and J. Pest, "Thermal feedback in power semiconductor devices," IEEE Trans. Electron Devices, vol. ED-17, pp. 770- 782, Sept., 1970.

[Nin84] T. Ning and D. Tang, "Method of determining the emitter and base resistance of bipolar transistors", IEEE Trans. Electron Devices., vol. ED-31, pp. 409-412, 1984.

[Oet76] F. F. Oettinger, D. L. Blackburn, and S. Rubin, "Thermal characterization of power transistors," IEEE Trans. Electron Devices, vol. ED-23, pp. 831-838, Aug. 1976.

[Rei92] M. Reisch, "Self-heating in BJT circuit parameter extraction," Solid-State Electronics, vol. 35, no. 5, pp. 677-679, May 1992.

[Sof94] J. W. Sofia, "Analysis of thermal transient data with synthesized dynamic models for semiconductor devices", Proc. Tenth IEEE Semi-Therm, pp. 78-85, Aug. 1994.

[Suh95] D. Suh, Modeling of Non-Fully Depleted Silicon-on-Insulator MOSFETs, and Applications to High-Performance/Low Power ULSI Design. Ph.D. Dissertation, University of Florida, Gainesville, 1995.

[Sze81] S. M. Sze, Physics of Semiconductor Devices. Second Edition, Wiley, New York, 1981.

[Szè88] V. Szèkely and T. Van Bien, "Fine structure of heat flow path in semiconductor devices: a measurement and identification method", Solid-State Electronics, vol. 31, no. 9, pp. 1363-1368, Sept. 1988.

[Vog89] R. S. Vogelsong and C. Brzezinski, "Extending SPICE for electrothermal simulation, Proc. IEEE Custom Integrated Circuits Conf., pp. 21.4.1-21.4.4, May 1989.

[Yeh95] P. C. Yeh and J. G. Fossum, "Physical subthreshold MOSFET modeling applied to viable design of deep-submicron fully depleted SOI low-voltage CMOS technology," IEEE Trans. Electron Devices, vol. 42, p. 1605, Sept. 1995.

[Wor96] G. O. Workman, S. Krishnan, J. G. Fossum, M. M. Pelella, "DC and transient temperature dependences in SOISPICE PD and FD MOSFET models," Proc. TECHCON 96, Aug. 1996.

[Zwe92] D. T. Zweidinger, S.-G. Lee, and R. M. Fox, "Compact modeling of BJT self-heating in SPICE," Proceedings of NUPAD IV, pp. 219-223, May 1992.

[Zwe93] D. T. Zweidinger, S.-G. Lee, and R. M. Fox, "Compact modeling of BJT self-heating in SPICE," IEEE Trans. Computer-Aided Design, pp. 1368-1375, Sept. 1993.

[Zwe95a] D. T. Zweidinger, R. M. Fox, S.-G. Lee, and T. Jung, "Extraction of thermal parameters for bipolar circuit simulation," Proc. IEEE Bipolar/BiCMOS Circuits and Technology Meeting, Oct. 1995.

[Zwe95b] D. T. Zweidinger, J. S. Brodsky, and R. M. Fox, "A physical thermal resistance model for vertical BJTs on SOI," IEEE International SOI Conf., Oct. 1995.

[Zwe96] D. T. Zweidinger, R. M. Fox, J. S. Brodsky, T. Jung, and S.-G. Lee "Thermal impedance extraction for bipolar transistors," IEEE Transactions on Electron Devices, vol. 43, no. 2, pp. 342-346, Feb. 1996.

BIOGRAPHICAL SKETCH

David Todd Zweidinger received the BSEE degree in 1990, specializing in computer engineering, from the University of Florida. He later received the Master of Science degree in electrical engineering in 1993 from the University of Florida for his thesis work on digital electronics and digital signal processing. He is currently completing the Ph.D degree in 1997 at the University of Florida studying thermal effects in semiconductor compact modeling. He is employed at Harris Semiconductor where he is continuing work on improving circuit simulation.

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a thesis for the degree of Doctor of Philosophy.

Robert M. Fox, Chairman
Associate Professor of Electrical and Computer Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a thesis for the degree of Doctor of Philosophy.

William R. Eisenstadt
Associate Professor of Electrical and Computer Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a thesis for the degree of Doctor of Philosophy.

Mark E. Law
Professor of Electrical and Computer Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a thesis for the degree of Doctor of Philosophy.

Jill Peterson
Associate Professor of Mechanical Engineering

This dissertation was submitted to the Graduate Faculty of the College of Engineering and to the Graduate School and was accepted as partial fulfillment of the requirements for the degree of Doctor of Philosophy.

August 1997

Winfred M. Phillips
Dean, College of Engineering

Karen A. Holbrook
Dean, Graduate School